AD-A100 816 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 12/1
AN INTERACTIVE PROGRAM FOR THE CALCULATION AND ANALYSIS OF THE --ETC(U)
MAR 81 L K PALMER
UNCLASSIFIED AFIT/GA/EE/81M-1 NL

1 OF 2
AD
A100816

AFIT/GA/EE/81M-1

AN INTERACTIVE PROGRAM FOR THE CALCULATION

AND ANALYSIS OF THE PARAMETER SENSITIVITIES

IN A LINEAR, TIME-INVARIANT SYSTEM

AFIT/GA/EE/81M-1

Linda K. Palmer
1st Lt USAF

Approved for public release; distribution unlimited

AFIT/GA/EE/81M-1

AN INTERACTIVE PROGRAM FOR THE CALCULATION AND ANALYSIS OF THE PARAMETER SENSITIVITIES IN A LINEAR, TIME-INVARIANT SYSTEM

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by
Linda K. Palmer, B. A.
1st Lt USAF
Graduate Astronautical Engineering

March 1981

Preface

In identifying and estimating the parameters of a control system via quasilinearization, the output "sentitivity matrix" is perhaps the most costly and time-consuming portion of the required calculation. A new and highly efficient algorithm for calculating the sensitivity matrix of a linear, time invariant control system is developed in this paper. It is limited to the single-input single-output case; however, it may easily be modified to handle several inputs and outputs. Also, the input must be piecewise constant; and the output measurements must be taken at constant time intervals.

Thanks are due to Dr. J. Gary Reid, who has researched the problem and who developed the basic algorithm; to Dr. David A. Lee, who helped me with the mathematics; and to Mrs. Shirley J. Rapozo, who assisted me with the typing. And, of course, special thanks are due to my husband, Leslie, without whose computer knowledge and moral support I would never have finished this paper.

Linda K. Palmer

Contents

List of Figures

List of Tables

Notation

Symbol	Meaning
A	system plant matrix
$\underline{B}$	system input vector
$\underline{C}$	system output vector
E	time-dependent decomposition of S
EIGV	column of eigenvectors; each column is a vector
$\underline{e}_k'$	kth row vector of E
E1	first half of E matrix
E3	second half of E matrix
F	input- and time-dependent decomposition of S
$\underline{f}_k'$	kth row vector of F
F1	first half of F matrix
F3	second half of F matrix
G	structure-dependent decomposition of S
G_{zi}	zero input part of G
G_{zs}	zero state part of G
g_{zi_i}	ith column of G_{zi}
H	matrix used to find eigenvector sensitivities
H.O.T.	higher order terms
$h^{\ell,m}_{k,j}$ and $h^{\ell,m}_{j,k}$	elements of matrix H
IA	matrix containing numbers and locations of parameters in A

Symbol	Meaning
IB	matrix containing numbers and locations of parameters in $\underline{B}$
IC	matrix containing numbers and locations of parameters in $\underline{C}$
IX	matrix containing numbers and locations of parameters in $\underline{x}_0$
K	number of sample times
NA	dimension of A
NB	dimension of B in multi-input multi-output case
NC	dimension of C in multi-input multi-output case
NP	total number of parameters
NPA	number of parameters in A
NPB	number of parameters in $\underline{B}$
NPC	number of parameters in $\underline{C}$
NPX	number of parameters in $\underline{x}_0$
R^{NP}	parameter space of dimension NP
REIGV	matrix of reciprocal eigenvectors; each column is a vector
S	output sensitivity matrix
t	time
t_f	final time
t_k	sample time
U	system input
U_d	discretized system input
UT_1	orthogonal matrix of left singular vectors
$\underline{u}_j$	jth eigenvector
V	orthogonal matrix of right singular vectors

Symbol	Meaning
V_2	columns of V corresponding to the zero diagonal elements of Σ
$\underline{v}_j$	jth column of V
$\underline{v}_j'$	jth reciprocal eigenvector
$\underline{x}$	state vector
$\underline{x}_o$	nominal value of state vector
$\underline{Y}$	vector of output response measured at each time t_k
$\underline{\hat{Y}}(\underline{\hat{\theta}}_o)$	vector of predicted output response
y	system output
Δ	sample spacing
δ,ε	vectors of negligible size
$\underline{\Gamma}$	$\underline{\hat{Y}} - \underline{\hat{Y}}(\underline{\hat{\theta}}_o)$
κ	condition number of a matrix $= \frac{\sigma_{max}}{\sigma_{min}} = \frac{\sigma_1}{\sigma_{NP}}$
λ_j	jth eigenvalue of A
η	non-negative, problem-dependent constant
σ_j	jth singular value of S (or G)
Σ	diagonal matrix of singular values
$\underline{\theta}$	parameter vector
$\underline{\hat{\theta}}_o$	best estimate of parameter vector
$\underline{\Delta\theta}$	$\underline{\theta} - \underline{\hat{\theta}}_o$
$\underline{\Delta\theta}^*$	update of old and current best estimate of $\underline{\theta}$
$\|\cdot\|$	spectral norm

subscripts

$(\ )_{(i)}$	partial differentiation with respect to θ_i
$(\ )_j$	related to jth value or vector
$(\ )_{zi}$	related to zero input response
$(\ )_{zs}$	related to zero state response
$(\underline{\ })$	vector quantity

superscripts

$(\ )'$	transpose
$(\ )^{-1}$	inverse
$(\ )^{\dagger}$	pseudo inverse
$(\hat{\ })$	estimated value

Abstract

In this paper, a new algorithm is developed to calculate the output "sensitivity matrix" of a linear, time-invariant, single-input single-output control system with piecewise constant input and output measurements taken at constant time intervals. The algorithm incorporates the singular value decomposition to investigate parameter identifiability and estimation accuracy in relation to the system as a whole and in relation to the model of the system. As a result, a structural condition on identifiability is imposed; and the system designer now has a tool to evaluate how well the model describes the system.

The algorithm is verified by checking its results with those using a standard software package for numerical integration. It is then used to investigate input and structural design issues.

I. Introduction

In analyzing a system, the design engineer uses a mathematical model. The model, by its very definition, represents the system. It must account for any variations in the system, such as changes in the initial state or in the input. The model must be refined for the uncertainties in the system's environment, such as temperature or imperfect measurements, and for the effect of higher order terms if they have been neglected. The parts of the system affected by these changes and uncertainties are called the system parameters.

In the process of system identification and parameter estimation, the model parameters are found and given values (Ref 1). This paper considers the state variable model, which is widely used since it requires the least amount of *a priori* information to predict the future (Ref 15:2). The state variable model is a set of linear, first-order differential equations. However, the relationship between the system's output and its input and model parameters is, in general, highly nonlinear. (See Eq (4).) Therefore, system identification and parameter estimation result in a nonlinear optimization problem, where convergence is desired between the output predicted by the model and the actual measured output of the system (Ref 15:3). Basically, the parameter identification and estimation problem may be solved by linearizing the predicted output response about the current best estimate of the parameters, equating the result to the actual measured outputs to get a parameter update, and repeating the process until the parameter update goes to zero. This process is called quasilinearization. For more details, see Eqs (1) through (11).

The most costly and time-consuming part of quasilinearization is the formation of the output "sensitivity matrix." (See Eq (8).) This paper is concerned with the formation of the sensitivity matrix using two methods. The first method involves the use of the "sensitivity system," a set of first-order differential equations, and is treated as the "standard" method. In the second method, the sensitivity matrix is decomposed into input-, time-, and structure-dependent parts, and may be referred to as a "modal" sensitivity approach. (See Eqs (33) through (59).) The sensitivity matrix may be decomposed because the system output may be expressed in terms of the system input; the input, output, and state vectors of the mathematical model; and the eigenvalues and eigenvectors of the system plant matrix. The reader must remember that the formation of the sensitivity matrix constitutes only a part of the identification/estimation problem and must be calculated at every iteration of the solution.

In Chapter I., the theory of the quasilinearization and the two algorithms for calculating the sensitivity matrix are explained; and the algorithms are evaluated. Solvability of the problem, or identifiability of the parameters, and the accuracy of the parameter estimation are discussed.

Chapter III contains the computational aspects of the new algorithm. This includes an explanation of the interactive program developed to implement the algorithm as well as a summary of the computational loading of the program. Also included are discussions of how the program may be used by the designer both in the experimental design phase and in the actual estimation task, and of the effect of variations on the basic assumptions of the algorithm.

The new algorithm is validated by comparing its result with that of the standard sensitivity system method in Chapter IV. Then the new algorithm is used to investigate both structural and input design issues of the model. Several examples are presented and analyzed.

Conclusions about the new algorithm and recommendations for further research are made in Chapter V.

II. Theory

Since the two procedures to be discussed are used in the technique of quasilinearization, the theory of this technique is first explained. The singular value decomposition and the insight it gives into identifiability and accuracy of parameter estimation are also discussed. Then the "sensitivity matrix" S of the state-variable model of a linear time-invariant control system is formed first, by the "sensitivity system" method and second, via the new "modal" method. The methods are compared, and a structural condition on identifiability is presented.

Quasilinearization (Ref 4; 6; 7; 10; 11; 13; 14)

The parameters of the system, which may have real or complex values, are assembled into a vector $\underline{\theta}$ of dimension NP. The single-input single-output system considered is of order NA and has state dynamics

$$\dot{\underline{x}}(t) = A(\underline{\theta})\ \underline{x}(t) + \underline{B}(\underline{\theta})\ U(t) \qquad t \in [0, t_f] \tag{1}$$

$$\underline{x}(0) = \underline{x}_0(\underline{\theta}) \tag{2}$$

with output measurements at each time t_k

$$y(t_k) = \underline{C}(\underline{\theta})\ \underline{x}(t_k)$$
$$t_k \in [0, t_f] \qquad k = 1, 2, \ldots, K \tag{3}$$

The system input U is assumed to be known exactly, and the nominal values of A, $\underline{B}$, $\underline{C}$, and $\underline{x}_0$ are assumed to be good approximations of their true values. The matrix A is assumed to be non-defective and to have NA distinct eigenvalues. All system matrices and vectors may be real or complex. Parameters θ_i, which may also

be either real or complex, are assumed to appear linearly in the matrix A and in the NA-dimensional vectors $\underline{B}$, $\underline{C}$, and $\underline{x}_0$. Each parameter may appear more than once in the plant matrix A and/or in the vectors $\underline{B}$, $\underline{C}$, and $\underline{x}_0$.

The output at time t_k may be calculated from the equation

$$y(t_k) = \underline{C}\, e^{At_k}\, \underline{x}_0 + \underline{C} \int_0^{t_k} e^{A(t_k - \tau)}\, \underline{B}\, U(\tau)\, d\tau$$

$$= \sum_{j=1}^{NA} (\underline{C}\, \underline{u}_j)\, (\underline{v}_j'\, \underline{x}_0)\, e^{\lambda_j t_k}$$

$$+ \sum_{j=1}^{NA} (\underline{C}\, \underline{u}_j)\, (\underline{v}_j'\, \underline{B})\, e^{\lambda_j t_k} \int_0^{t_k} e^{-\lambda_j \tau}\, U(\tau)\, d\tau \qquad (4)$$

where $\underline{u}_j$ is the jth eigenvector, and $\underline{v}_j'$ is the jth reciprocal eigenvector. (See, for example, Reference 12, page 3.) These vectors correspond to the jth eigenvalue λ_j.

If $\hat{\underline{\theta}}_0$ is the current best estimate of the unknown parameter vector, then the predicted output response may be written as

$$\hat{\underline{Y}}(\hat{\underline{\theta}}_0) = \begin{bmatrix} y(t_1, \hat{\underline{\theta}}_0) \\ y(t_2, \hat{\underline{\theta}}_0) \\ \vdots \\ y(t_K, \hat{\underline{\theta}}_0) \end{bmatrix}_{K \times 1} \qquad (5)$$

Linearizing Eq (5) about $\underline{\hat{\theta}}_o$ and equating to the actual measured outputs yields

$$\begin{bmatrix} y(t_1) \\ y(t_2) \\ \cdot \\ \cdot \\ \cdot \\ y(t_K) \end{bmatrix} = \begin{bmatrix} (y_1, \underline{\hat{\theta}}_o) \\ (y_2, \underline{\hat{\theta}}_o) \\ \cdot \\ \cdot \\ \cdot \\ (y_K, \underline{\hat{\theta}}_o) \end{bmatrix} + \begin{bmatrix} y_{(1)}(t_1) & y_{(2)}(t_1) & \cdots & y_{(NP)}(t_1) \\ y_{(1)}(t_2) & y_{(2)}(t_2) & \cdots & y_{(NP)}(t_2) \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ y_{(1)}(t_K) & y_{(2)}(t_K) & \cdots & y_{(NP)}(t_K) \end{bmatrix} \begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \\ \cdot \\ \cdot \\ \cdot \\ \Delta\theta_{NP} \end{bmatrix}$$

$$+ \text{ H.O.T.} + \text{Noise} \qquad (6)$$

Ignoring higher-order terms and noise, Eq (6) may be written in matrix-vector form as

$$\underline{\Gamma} = \left[\underline{Y} - \underline{\hat{Y}}(\underline{\hat{\theta}}_o) \right] = S\ \underline{\Delta\theta} \qquad (7)$$

The sensitivity matrix S in Eq (7) is defined by

$$S = \begin{bmatrix} y_{(1)}(t_1) & y_{(2)}(t_1) & \cdots & y_{(NP)}(t_1) \\ y_{(1)}(t_2) & y_{(2)}(t_2) & \cdots & y_{(NP)}(t_2) \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ y_{(1)}(t_K) & y_{(2)}(t_K) & \cdots & y_{(NP)}(t_K) \end{bmatrix}_{K \times NP} \qquad (8)$$

where

$$y_{(i)}(t_k) = \left. \frac{\partial y(t_k)}{\partial \theta_i} \right|_{\theta = \hat{\theta}_o} \qquad (9)$$

for k=1, 2, . . . ,K and i=1, 2, . . . , NP (Ref 15: 3).

The best approximation of $\underline{\Delta\theta}$ is obtained by finding $\underline{\Delta\theta}^*$, the $\underline{\Delta\theta}$ which minimizes

$$\left\| \underline{\Gamma} - S\ \underline{\Delta\theta} \right\|^2 \qquad (10)$$

If S has rank NP (where the number of samples K is assumed to be greater

than or equal to the number of parameters NP), then a unique solution $\underline{\Delta\theta}^*$ exists. The use of singular value decomposition to find the rank of S is discussed in the next section of this chapter. The calculation of $\underline{\Delta\theta}^*$ from the singular value decomposition is also discussed in the next section.

Once $\underline{\Delta\theta}^*$ has been obtained, the parameter vector may be updated by

$$\hat{\underline{\theta}}_{o_{new}} = \hat{\underline{\theta}}_{o_{old}} + \underline{\Delta\theta}^* \tag{11}$$

The process is repeated until convergence is obtained, or until $\underline{\Delta\theta}^*$ goes to $\underline{0}$.

Singular Value Decomposition

Earlier in this chapter, it was stated that S must have rank NP for $\underline{\Delta\theta}^*$ to minimize Eq (10)(Ref 17: 634-635). Readily available software packages may be used to obtain the singular value decomposition of the matrix S (Ref 5: LSVDF), making the singular value decomposition a reliable and easily implemented method of calculating the rank of S. The singular value decomposition also gives insight into the accuracy of the parameter estimation and into the "directions" of best identification, as will be explained in the following sections.

The Method. If the matrix S has rank NP, assuming $NP \leq K$, then S will have NP nonzero singular values $\sigma_1, \sigma_2, \ldots, \sigma_{NP}$ (Ref 17: 637). Then S will have the singular value decomposition

$$S_{K \times NP} = [UT_1 \quad UT_2]_{K \times K} \begin{bmatrix} \Sigma \\ 0 \end{bmatrix}_{K \times NP} [V]'_{NP \times NP}$$

$$= UT_1 \; \Sigma \; V' \tag{12}$$

where Σ is the diagonal matrix of singular values such that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{NP} \tag{13}$$

(Ref 5: LSVDF). The orthogonal matrices UT_1 and V are composed of the left the right singular vectors of S, respectively. The unique vector $\underline{\Delta\theta}^*$ is then given by

$$\underline{\Delta\theta}^* = V \; \Sigma^{-1} \; UT_1 \; \underline{\Gamma} \tag{14}$$

See Stewart (Ref 17: 636) for mathematical details.

Since Σ is diagonal, Σ^{-1} is formed by taking the multiplicative inverses of each of the elements along the diagonal of Σ. If Σ is not invertible, then some of the singular values of S are identically zero. Then V_2, the columns of V corresponding to the zero singular values, span the null space of S, and the unique vector $\underline{\Delta\theta}^*$ minimizing Eq (10) does not exist. This means that not all of the parameters of Eqs (1) through (3) can be identified, and the system is termed "nonidentifiable" (Ref 15: 10). When this occurs, the system must be modified either by eliminating or combining several of the unknown parameters or by remodelling the system itself (Ref 15: 5-7) so that S has rank NP.

Accuracy of the Estimation. Once the parameters have been estimated, a measure of the quality of the estimation is desired. To find a suitable measure, Stewart derives the following spectral norm relations (Ref 17: 636)

$$\|S\| = \sup \frac{\|S \, \underline{\Delta\theta}\|}{\|\underline{\Delta\theta}\|} = \sigma_1 = \sigma_{max} \tag{15}$$

$$\|S^{-}\| = \sup \frac{\|S \; \underline{\Gamma}\|}{\|\underline{\Gamma}\|} = \sigma_{NP} = \sigma_{min} \tag{16}$$

where $\|\cdot\|$ denotes the spectral norm and S is the pseudo-inverse of S, defined by

$$S^{+} S S^{+} = S^{+} \tag{17}$$

and

$$S S^{+} S = S \tag{18}$$

(Ref 17: 634). Then, using Eq (12), S^{+} may be found to be equal to

$$S^{+} = V \Sigma^{-1} UT_1 \tag{19}$$

According to Stewart, the solution to the perturbed problem

$$\underline{\Gamma} + \underline{\delta} = S (\underline{\Delta\theta} + \underline{\varepsilon}) \tag{20}$$

satisfies the error bounds

$$\frac{\|\varepsilon\|}{\|\Delta\theta\|} \leq \kappa \eta \frac{\|\delta\|}{\|\Gamma_1\|} \tag{21}$$

where Γ_1 is part of the partitioned vector

$$\underline{\Gamma} = [\Gamma_1 \; \Gamma_2] \tag{22}$$

and where κ, the ratio of the largest to the smallest singular value of S, is called the condition number of S

$$\kappa = \frac{\sigma_1}{\sigma_2} \tag{23}$$

The condition number is bounded by unity from below and infinity from above. The nonnegative constant η is defined by

$$\|\Gamma_1\| = \eta \sigma_1 \|\underline{\Delta\theta}\| \tag{24}$$

and is problem-dependent. It may be shown that

$$\frac{1}{\kappa} \leq \eta \leq 1 \tag{25}$$

When S is ill-conditioned, i. e., when S^{+} is large, then η may be close to κ^{-1} (and both are near zero). Then $\underline{\Gamma}$ is said to "reflect the ill-

condition of S." If, however, η is near unity, then $\underline{\Gamma}$ does not reflect the ill-condition of S. Then the estimation accuracy is proportional to κ or to some power of κ, making the condition number a convenient and reliable measure of the condition of S (Ref 17: 653-655). No matter what η is, however, κ will always give a worst-case bound on the error: The closer κ is to unity, the better is the parameter estimation. When κ is not close to unity, the accuracy of the estimation may be questionable; and perhaps the model should be changed.

"Directions" of Best Identification. In the singular value decomposition defined by Eq (12), the NP columns of V, $\underline{\upsilon}_j$, j=1, 2,. . .,NP, form an orthogonal basis for the parameter space R^{NP} such that

$$\|S \underline{\upsilon}_j\|^2 = \underline{\upsilon}_j' (S'S) \underline{\upsilon}_j = \sigma_j^2 \tag{26}$$

Since the singular values are ordered from largest to smallest, $\underline{\upsilon}_1$ gives the "direction" of best identification; while $\underline{\upsilon}_{NP}$ gives the "direction" of worst identification. This means that the first parameters, corresponding to the larger singular values, will be easier to identify and estimate accurately than the last parameters, which correspond to the smaller singular values. If the estimate is too inaccurate, the designer might try to reduce the model, thereby reducing the number of unknown parameters (Ref 8).

The next two sections of this chapter are concerned with calculating the sensitivity matrix.

Sensitivity System Calculation of S

The "standard" sensitivity system method of calculating the sensitivity matrix is described in this section. For ease of notation, functions of

$\underline{\theta}$, e. g., $A \equiv A(\underline{\theta})$, are implicit. In addition, $\underline{x}$ and U are functions of t. Taking the partial derivatives of Eqs (1) and (3) with respect to the ith parameter yields

$$\dot{\underline{x}}_{(i)} = A\,\underline{x}_{(i)} + A_{(i)}\,\underline{x} + \underline{B}_{(i)}\,U \tag{27}$$

$$y_{(i)}(t_k) = \underline{C}\,\underline{x}_{(i)}(t_k) + \underline{C}_{(i)}\,\underline{x}(t_k) \tag{28}$$

for i= 1, 2,. . ., NP and k= 1, 2,. . ., K. The sensitivity system becomes

$$\begin{bmatrix} \dot{\underline{x}} \\ \dot{\underline{x}}_{(1)} \\ \vdots \\ \dot{\underline{x}}_{(NP)} \end{bmatrix} = \begin{bmatrix} A & 0 & \cdots & 0 \\ A_{(1)} & A & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ A_{(NP)} & 0 & \cdots\ 0 & A \end{bmatrix} \begin{bmatrix} \underline{x} \\ \underline{x}_{(1)} \\ \vdots \\ \underline{x}_{(NP)} \end{bmatrix} + \begin{bmatrix} \underline{B} \\ \underline{B}_{(1)} \\ \vdots \\ \underline{B}_{(NP)} \end{bmatrix} U \tag{29}$$

where the new "plant matrix" is formed by placing A along the diagonal, the partial derivatives of A along the first column, and zero matrices elsewhere. Similarly, the new "output" equation becomes

$$\begin{bmatrix} y \\ y_{(1)} \\ \vdots \\ y_{(NP)} \end{bmatrix} = \begin{bmatrix} \underline{C} & 0 & \cdots & 0 \\ \underline{C}_{(1)} & \underline{C} & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ \underline{C}_{(NP)} & 0 & \cdots\ 0 & \underline{C} \end{bmatrix} \begin{bmatrix} \underline{x} \\ \underline{x}_{(1)} \\ \vdots \\ \underline{x}_{(NP)} \end{bmatrix} \tag{30}$$

for each time t_k= 1, 2,. . ., K. The differential equations (See Eq (29).) may be solved using ODE (Ref 10) or a similar software package.

The difficulty with this method is that it results in at most NA * (NP + 1) and at least NA * (NPA + NPB + NPX + 1 - number of parameters common to A, $\underline{B}$ and $\underline{x}_0$) coupled, linear differential equations which may be "stiff" and, therefore, hard to integrate accurately. This occurs when

the eigenvalues of the plant matrix A are far apart. An example of a very stiff system is the two-dimensional case (NA= 2) where the eigenvalues λ_1 and λ_2 equal -1 and -100, respectively.

Another shortcoming of this method is its "uncontrollability," occurring when the number of equations in the system is too large. Unnecessary work is created in a direct integration of the excess equations of the system. To lessen the workload, model order reduction techniques may be used to reduce the number of equations from NA * (NP + 1) to about 2 NA for a single-input single-output system (or to about 2 NA * NB, where NB is the number of inputs of the control system) (Ref 8). Unfortunately, the linear transformations required by the model order reduction techniques may significantly increase the error.

In addition, this method gives no indication about how well the model described by Eqs (1) through (3) actually represents the control system. The desirability of such "structural" insight leads naturally to the decomposition of S into time-, input- and structure-dependent parts.

Modal Algorithm for Calculation of S

In this section, a new algorithm for calculating the sensitivity matrix is developed. Because the output sensitivities are expressed in terms of the eigenvalues and eigenvectors of A, this method may be called the "modal" approach:

From linear systems theory, when the NA-dimensional matrix A has NA distinct eigenvalues

$$e^{At_k} = \sum_{j=1}^{NA} \underline{u}_j \, e^{\lambda_j t_k} \, \underline{v}_j' \qquad (31)$$

Noting Eq (31) and taking the partial derivatives of all the terms in Eq (4) with respect to θ_i gives

$$y_{(i)}(t_k) = \sum_{j=1}^{NA} [\, \underline{C}_{(i)}\, \underline{u}_j\, e^{\lambda_j t_k}\, \underline{v}_j'\, \underline{x}_o + \underline{C}\, \underline{u}_{j(i)}\, e^{\lambda_j t_k}\, \underline{v}_j'\, \underline{x}_o$$

$$+ \underline{C}\, \underline{u}_j\, e^{\lambda_j t_k}\, \underline{v}_{j(i)}'\, \underline{x}_o + \underline{C}\, \underline{u}_j\, e^{\lambda_j t_k}\, \underline{v}_j'\, \underline{x}_{o(i)}$$

$$+ \underline{C}_{(i)}\, \underline{u}_j\, e^{\lambda_j t_k}\, \underline{v}_j'\, \underline{B} + \underline{C}\, \underline{u}_{j(i)}\, e^{\lambda_j t_k}\, \underline{v}_j'\, \underline{B}$$

$$+ \underline{C}\, \underline{u}_j\, e^{\lambda_j t_k}\, \underline{v}_{j(i)}'\, \underline{B} + \underline{C}\, \underline{u}_j\, e^{\lambda_j t_k}\, \underline{v}_j'\, \underline{B}_{(i)}$$

$$* \int_0^{t_k} e^{-\lambda_j \tau}\, U(\tau)\, d\tau \,] \qquad (32)$$

Calculating $y_{(i)}(t_k)$ for i = 1, 2,. . ., NP and k = 1, 2,. . ., K gives the elements of the K x NP sensitivity matrix defined by Eq (8). By rearranging the terms on the right-hand side of Eq (32) for all i and k, and by expressing the result in matrix-vector form, the sensitivity matrix may be written

$$S = [E \quad F] \;\; G \qquad (33)$$

where E depends on time alone, F is dependent on both time and input, and G depends solely on the structure of the system matrix and vectors (Ref 15: 4). The matrices E and F have row vectors

$$\underline{e}_k' = \left[e^{\lambda_1 t_k},\; e^{\lambda_2 t_k},\; \ldots,\; e^{\lambda_{NA} t_k},\; t_k\, e^{\lambda_1 t_k},\; t_k\, e^{\lambda_2 t_k},\; \ldots,\; t_k\, e^{\lambda_{NA} t_k} \right]_{1 \times 2NA} \qquad (34)$$

$$\underline{f}_k' = \left[e^{\lambda_1 t_k} \int_0^{t_k} e^{-\lambda_1 \tau} U(\tau)\, d\tau \ , \ e^{\lambda_2 t_k} \int_0^{t_k} e^{-\lambda_2 \tau} U(\tau)\, d\tau \ , \right.$$

$$\ldots, \ e^{\lambda_{NA} t_k} \int_0^{t_k} e^{-\lambda_{NA} \tau} U(\tau)\, d\tau \ , \ e^{\lambda_1 t_k} \int_0^{t_k} (t_k - \tau)$$

$$* \ e^{-\lambda_1 \tau} U(\tau)\, d\tau \ , \ e^{\lambda_2 t_k} \int_0^{t_k} (t_k - \tau)\, e^{-\lambda_2 \tau} U(\tau)\, d\tau \ ,$$

$$\left. \ldots, \ e^{\lambda_{NA} t_k} \int_0^{t_k} (t_k - \tau)\, e^{-\lambda_{NA}} U(\tau)\, d\tau \right]_{1 \times 2NA} \qquad (35)$$

If U is piecewise constant and if the sample times are evenly spaced, then $\underline{e}_k'$ and $\underline{f}_k'$ may each be divided into two parts

$$\underline{e}_k' = [E1(k,\ell) \quad E3(k,\ell)] \qquad (36)$$

$$\underline{f}_k' = [F1(k,\ell) \quad F3(k,\ell)] \qquad \ell = 1, 2, \ldots, NA \qquad (37)$$

where the elements of $\underline{e}_k'$ and $\underline{f}_k'$ are given by the formulas

$$E1(k,\ell) = e^{\lambda_\ell \Delta} * E1(k-1,\ell) \qquad (38)$$

$$E3(k,\ell) = k * \Delta * E1(k,\ell) \qquad (39)$$

with

$$E1(1,\ell) = e^{\lambda_\ell \Delta} \qquad (40)$$

$$E3(1,\ell) = \Delta e^{\lambda_\ell \Delta} = \Delta * E1(1,\ell) \qquad (41)$$

and

$$F1(k,\ell) = e^{\lambda_\ell \Delta} * \ F1(k-1,\ell) + C4 * U_d(k) \qquad (42)$$

$$F3(k,\ell) = e^{\lambda_\ell \Delta} * [F3(k-1,\ell) + \Delta * F1(k-1,\ell) + C5 * U_d(k)] \qquad (43)$$

with

$$F1(1,\ell) = C4 * U_d(1) \tag{44}$$

$$F3(1,\ell) = C5 * U_d(1) \tag{45}$$

where Δ equals the sample spacing, $C4 = (e^{\lambda_\ell \Delta} - 1)/\lambda_\ell$. and $C5 = (1 + \Delta\lambda_\ell * e^{\lambda_\ell \Delta} - e^{\lambda_\ell \Delta})/\lambda_\ell^{\,2}$. See Appendix A for the derivation of Eqs (38) through (45). Note that since λ_ℓ appears in the denominators of constants such as C4 and C5 (See Subroutine EFMAT in Appendix B.), all the eigenvalues of A must be nonzero.

The input- and time-dependent matrix [E F] has dimension K x 4NA, where K is the number of samples taken. This matrix is multiplied in Eq (33) to the 4NA x NP structure-dependent matrix

$$G = \begin{bmatrix} G_{zi} \\ G_{zs} \end{bmatrix}_{4NA \times NP} \tag{46}$$

where zi stands for zero input and zs stands for zero state. The ith column of G_{zi} is given by

$$g_{zi\,i} = \begin{bmatrix} \frac{\partial}{\partial \theta_i}[(\underline{C}\ \underline{u}_1)\ (\underline{v}_1'\ \underline{x}_o)] \\ \vdots \\ \frac{\partial}{\partial \theta_i}[(\underline{C}\ \underline{u}_{NA})\ (\underline{v}_{NA}'\ \underline{x}_o)] \\ \frac{\partial \lambda_1}{\partial \theta_i}[(\underline{C}\ \underline{u}_1)\ (\underline{v}_1'\ \underline{x}_o)] \\ \vdots \\ \frac{\partial \lambda_1}{\partial \theta_i}[(\underline{C}\ \underline{u}_{NA})\ (\underline{v}_{NA}'\ \underline{x}_o)] \end{bmatrix}$$

$$= \begin{bmatrix} [\underline{C}_{(i)} \underline{u}_1 + \underline{C} \underline{u}_{1(i)}] (\underline{v}_1' \underline{x}_o) + (\underline{C} \underline{u}_1) [\underline{v}_{1(i)}' \underline{x}_o + \underline{v}_1' \underline{x}_{o(i)}] \\ \vdots \\ [\underline{C}_{(i)} \underline{u}_{NA} + \underline{C} \underline{u}_{NA(i)}] (\underline{v}_{NA}' \underline{x}_o) + (\underline{C} \underline{u}_{NA}) [\underline{v}_{NA(i)}' \underline{x}_o + \underline{v}_{NA}' \underline{x}_{o(i)}] \\ \lambda_{1(i)} (\underline{C} \underline{u}_1) (\underline{v}_1' \underline{x}_o) \\ \vdots \\ \lambda_{2(i)} (\underline{C} \underline{u}_{NA}) (\underline{v}_{NA}' \underline{x}_o) \end{bmatrix} \tag{47}$$

The ith column of G_{zs} is the same as that of G_{zi} except that $\underline{x}_o$ is replaced by $\underline{B}$ (Ref 15: 4, 5). At this point $\underline{B}$, $\underline{C}$, $\underline{x}_o$, $\underline{u}_j$ and $\underline{v}_j'$ are known. The sensitivities of the quantities necessary to compute G_{zi} and G_{zs} are discussed in the next two sections.

Eigenvalue and Eigenvector Sensitivities. Crossley and Porter have derived closed form expressions for the sensitivities of λ_j , $\underline{u}_j$ and $\underline{v}_j'$ with respect to parameters in A, provided all λ_j of A are distinct (Ref 2: 163-170). Their results, in the terminology of this paper, follow. By definition A, λ_j , $\underline{u}_j$ and $\underline{v}_j'$ satisfy the equations

$$A \underline{u}_j = \lambda_j \underline{u}_j \qquad j = 1, 2, \ldots, NA \tag{48}$$

$$\underline{v}_j' A = \underline{v}_j' \lambda_j \tag{49}$$

$$\underline{v}_m' \underline{u}_\ell = \underline{u}_\ell' \underline{v}_m = \delta_{\ell m} \qquad \ell, m = 1, 2, \ldots, NA \tag{50}$$

where $\delta_{\ell m}$ is the Kronecker delta:

$$\delta_{\ell m} = \begin{cases} 1 & \text{if } \ell = m \\ 0 & \text{if } \ell \neq m \end{cases} \tag{51}$$

Differentiation of Eq (48) with respect to θ_i and premultiplication by $\underline{v}_j'$ yields

$$\underline{v}_j' A_{(i)} \underline{u}_j + \lambda_j \underline{v}_j' \underline{u}_{j_{(i)}} = \underline{v}_j' \lambda_{j_{(i)}} \underline{u}_j + \lambda_j \underline{v}_j' \underline{u}_{j_{(i)}}$$

$$i = 1, 2, \ldots, NPA \tag{52}$$

where NPA equals the number of parameters in A. Solving Eq (52) for $\lambda_{j_{(i)}}$ yields

$$\lambda_{j_{(i)}} = REIGV(\ell,j) * EIGV(m,j) \tag{53}$$

where

$$REIGV = \left[\underline{v}_1 \; \underline{v}_2 \; \cdot \; \cdot \; \cdot \; \underline{v}_{NA} \right] \tag{54}$$

$$EIGV = \left[\underline{u}_1 \; \underline{u}_2 \; \cdot \; \cdot \; \cdot \; \underline{u}_{NA} \right] \tag{55}$$

and θ_i appears in the ℓ,m location of A.

Through similar operations on Eqs (48) and (49) and by introducing a matrix H with components

$$h_{k,j}^{\ell,m} = \frac{REIGV(\ell,k) * EIGV(m,j)}{\lambda_j - \lambda_k} \tag{56}$$

$$h_{k,j}^{\ell,m} = \frac{REIGV(\ell,j) * EIGV(m,k)}{\lambda_j - \lambda_k} \tag{57}$$

the eigenvector and reciprocal eigenvector sensitivities may be written

$$\underline{u}_{j(i)} = \sum_{\substack{k=1 \\ k \neq j}}^{NA} h_{k,j}^{\ell,m} * \underline{u}_k \tag{58}$$

$$\underline{v}'_{j(i)} = \sum_{\substack{k=1 \\ k \neq j}}^{NA} h_{j,k}^{\ell,m} * \underline{v}'_k \tag{59}$$

Other Vector Sensitivities. Since it is assumed that θ_i appears linearly in $\underline{B}$, $\underline{C}$ and $\underline{x}_0$, the partial derivatives of these vectors with respect to θ_i contain all zero elements except in the location of θ_i , where unity appears. The computational aspects of this assumption are discussed in the next chapter.

Singular Value Decomposition and G. At this point, all the equations necessary for the calculation of S have been derived. It can be shown that in order for S to have rank NP (a necessary condition for a solution of the identification/estimation problem to exist), both $\begin{bmatrix} E & F \end{bmatrix}$ and G must have rank NP (Ref 11: 244; 18: 91). The requirement on the ranks of these matrices leads directly to time, input, and structural conditions on identifiability. Only the structural condition on identifiability, i. e., that G have rank NP, is considered in the following discussion.

The rank condition on G may be shown to be satisfied by performing a singular value decomposition on G. As in Eq (10), the number of samples K equals or exceeds the total number of parameters NP. Then there will be NP singular values if G has rank NP. The singular value decomposition of G also provides very important information on the "structural" aspects of the problem: Calculating the condition number of G (See Eq (23).) can

give the designer a good indication of just how well the model defined by Eqs (1) through (3) describes the system. This capability is important, for the parameter estimation can only be as good as the model. Also, if the condition number of G indicates that the model is good, but the condition number of S shows that the estimation is not, the designer knows to change either the sample spacing and/or the input and number of samples to improve the estimation. A more detailed description of this use of the condition number in the design phase is presented in the next section.

Points for the Modal Method in the Design Phase

The design phase of the parameter identification/estimation problem involves evaluating the sensitivity matrix and, if necessary, changing some part of the total system (i. e., input, sample spacing and system model).

In order to calculate the sensitivity matrix via the sensitivity system method, the designer must form the (NA + 1) * (NP + 1) equations of the sensitivity system (Eqs (29) and (30)) from the system equations (Eqs (1) through (3)) each time a design change is made. Although the integration is made simple by using available software packages, setting up the sensitivity system can be a tedious chore. Using the interactive computer program which implements the modal algorithm, however, all the designer need do is input the matrices and vectors of Eqs (1) through (3) and view the results as the computer prints them out. If the results are less than desirable, the matrices and vectors are easily changed without having to rewrite the program.

In addition, the sensitivity system method gives no indication of how well the model describes the system. By contrast, in the modal algorithm

the time- and input-dependent parts are separate from that part of the sensitivity matrix which depends on the model. See Eq (33). Because of this, the designer can look at the condition number of G and decide whether the model is adequate. If he decides it is, but the sensitivity matrix is too ill-conditioned, the designer may conclude that some aspect of the input (sample spacing Δ or number of sample times K) is at fault and amend the situation. Since the program listed in Appendix B is interactive, any parts of the system are easily changed--without having to make up a new set of differential equations. Examples are presented in Chapter IV.

Summary

Both the sensitivity system method and the modal method of calculating the sensitivity matrix S of a linear, time-invariant control system may be used in identifying and estimating system parameters via quasilinearization. The theory of quasilinearization and the two methods of calculating S have been explained, and some of the advantages of the modal method over the sensitivity system method have been noted.

The singular value decomposition and the insights it gives into system idencification and parameter estimation have been touched upon. In addition, a structural condition on identifiability has been formed by combining the technique of singular value decomposition with the modal method.

From an analysis viewpoint, the new algorithm seems superior to the sensitivity system method. What remains to be seen is how easily the algorithm may be implemented on the computer. The computational load is the subject of the next chapter.

III. Computations

The modal algorithm for calculating the output sensitivity matrix was developed in the last chapter, and a listing of the interactive program implementing the algorithm appears in Appendix B. The interactive program proves useful to the system designer in both the experimental design phase and in the estimation phase of the system identification/estimation task. Using the interactive program in the experimental design phase, the designer may analyze alternative models, alternative sample spacings and alternative inputs using experimental data before the actual parameter estimation task. During the second phase, the estimation phase, the computational efficiencies of the program play a fundamental role because in the iterative quasilinearization technique (explained in Chapter II) the sensitivity matrix must be calculated at each iteration.

Some of the computational efficiencies of the modal algorithm are noted in the first section of this chapter. In the second section, the program implementing the modal algorithm is explained; and its computational loading is tabulated. Then variations of the basic assumptions of the algorithm, such as the nonlinear appearance of the parameters in the system vectors $\underline{B}$, $\underline{C}$, and $\underline{x}_0$, and their effect on the program are discussed.

Computational Efficiencies of the Modal Algorithm

In this section some of the computational efficiencies of the computer program listed in Appendix B are summarized.

The modal algorithm uses exponentials, multiplications, additions, and transfers from one matrix to another. The program takes advantage of

the sparsity of the sensitivity matrices and vectors such as $EIGV_{(i)}$ and $\underline{C}_{(i)}$. Rather than multiplying zero as well as nonzero matrix or vector elements, the computer program calculates only values affected by the parameters. For example, since θ_i is assumed to appear linearly in $\underline{B}$, $\underline{C}$ and $\underline{x}_o$, the sensitivity vectors $\underline{B}_{(i)}$, $\underline{C}_{(i)}$ and $\underline{x}_{o(i)}$ have all zero elements except in the location of θ_i, where unity appears. Therefore, scalars such as $\underline{C}_{(i)}\ \underline{u}_j$ do not require the multiplication of two vectors, but involve simply a transfer of the appropriate element from the vector $\underline{u}_j$ to the corresponding location in $\underline{C}_{(i)}\ \underline{u}_j$. This example is discussed further later in this chapter. The transfers just mentioned decrease computation time and do not cause roundoff errors. Another case in which advantage is taken of sparse matrices occurs when $\underline{x}_o$ is the zero vector: Computation time is decreased by not multiplying any vectors or matrices associated with $\underline{x}_o$. Instead of using Eq (33) to calculate S, since G_{zi} is identically zero when $\underline{x}_o$ is $\underline{0}$ (See Eq (47).), S becomes $F * G_{zs}$.

Another "high-quality" feature of the program is the use of readily available software packages to compute the eigenvalues and vectors (Ref 5: EIGRF) and the singular values and vectors (Ref 5: LSVDF). The singular value decomposition package requires that the matrix to be decomposed be real (Ref 5: LSVDF). However, the matrix to be decomposed (G_{zs}, G or S) is expressed in complex notation. Therefore, the vector of eigenvalues is checked. This entails checking NA rather than 2 NA * NP (for G_{zs}), 4 NA * NP (for G) or K * NP (for S) values. This check will be explained in the next section.

The Program

In this section the program implementing the modal algorithm will be explained. Stress is placed on the interaction between the user and the

computer: The user inputs information; and the computer calculates with the information, stores it, asks for more, prints out results, and so on. Once the program has been called, the user has the option of seeing an explanation of the matrices and vectors used in the algorithm. If the user opts not to see the explanation, he will be prompted for the necessary data to calculate first, the structure-dependent part and second, the input- and time-dependent parts of S.

Structure-Dependent Part of S. The user inputs the dimension of A (NA); the number of parameters in A (NPA), $\underline{x}_0$ (NPX), $\underline{B}$ (NPB), and $\underline{C}$ (NPC); and the total number of system parameters (NP). The user is then asked if $\underline{x}_0$ equals $\underline{0}$. If it does, the computer sets $\underline{x}_0$ to $\underline{0}$; if not, the computer asks for $\underline{x}_0$. The system plant matrix, input and output vectors and the matrices storing the numbers and locations of parameters in A, $\underline{x}_0$, $\underline{B}$ and $\underline{C}$ are requested and entered. The matrices are read in by columns, as mentioned earlier. The matrices containing the parameter numbers and locations are IA, IB, IC and IX. An example of the contents of IA follows: If θ_i appears in the (ℓ,m) location of the A matrix, the parameter number i is stored in location IA(1,LPA), LPA= 1, 2,. . ., NPA. The row address ℓ and column address m of the parameter are stored in locations IA(2,LPA) and IA(3,LPA), respectively. For a single-input single-output system, $\underline{B}$ and $\underline{C}$ are one-dimensional arrays. Therefore, IB and IC, as well as IX, have dimension 2 x NPB, NPC or NPX. The parameter numbers are stored in the first row, and the vector locations of the parameters are stored in the second row of IB, IC and IX.

IMSL subroutine EIGRF (EIGCC for complex A) computes the eigenvalues and eigenvectors of A. The eigenvalues are stored in the vector EIG, and

the eigenvectors are stored by columns in the matrix EIGV, as in Eq (55). The program then checks for repeated, zero and complex eigenvalues. If A has a repeated eigenvalue or an eigenvalue equal to zero, the program stops; and the user must remodel the system model so that A has NA distinct, nonzero eigenvalues. The matrix A must have distinct eigenvalues, or the denominators in Eqs (56) and (57) will be zero. The eigenvalues must be nonzero, or the denominators of the constants C4 and C5 will be zero. See Eqs (42) through (45). The effect of the check for complex eigenvalues is explained in the discussion of IMSL subroutine LSVDF later in this section.

To form the matrix of reciprocal eigenvectors, IMSL subroutine LEQT1C is called to invert the transpose of the matrix of eigenvectors. Each column of the matrix REIGV is a reciprocal eigenvector, as in Eq (54). Then $\underline{v}_j' \underline{x}_o$, $\underline{v}_j' \underline{B}$ and $\underline{C}' \underline{u}_j$, the vectors which do not depend on the parameters are formed. The user may choose to see these if he wishes.

The program then calls its own subroutine SENS to form the sensitivities of λ_j, u_j, v_j', x_o, $\underline{B}$ and $\underline{C}$ with respect to each parameter α_i. Since α_i may appear more than once in the system matrix and vectors, the sensitivities with respect to α_1 are formed first; then sensitivities with respect to α_2, α_3, and so on to α_{NP} are formed. This involves checking the matrices IA, IB, IC and IX, which hold the number and locations of each parameter, at most 2 * NP times. Since the sensitivity vectors $\underline{x}_{o(i)}$, $\underline{B}_{(i)}$, and $\underline{C}_{(i)}$ have zero or unity elements, the values of $\underline{u}_j$ and $\underline{v}_j$ corresponding to unity elements in the sensitivity vectors are transferred to the appropriate locations of $\underline{C}'_{(i)} u_j$, $v_j' x_{o(i)}$ and $\underline{v}_j' B_{(i)}$. The quantities $\lambda_{j(i)}$, $u_{j(i)}$ and $v'_{j(i)}$ are computed via methods proposed by

Crossley and Porter as described in Chapter II. When the plant matrix A is in the diagonal canonical form and the parameters are the elements along the main diagonal of A, the eigenvalues of A are the parameters themselves; and both EIGV and REIGV equal the identity matrix. Therefore, subroutine SENS could be simplified for the special case of the diagonal canonical form. But this is a topic to be left to further research.

If $\underline{x}_0$ is the zero vector, G_{zi} is identically zero; and subroutine GMATR is called to form G_{zs}. If $\underline{x}_0$ is not the zero vector, subroutine GMATR forms G_{zi} and then G_{zs}. G is then formed as in Eq (47). As may be seen from this equation, G is complex if any eigenvalues or eigenvectors of A or any of the system vectors $\underline{x}_0$, $\underline{B}$ or $\underline{C}$ are complex. This is the point at which the eigenvalue check mentioned earlier enters; for in order to perform a singular value decomposition on a complex matrix, it must be put into a real number format (Ref 5: LSVDF). Therefore, if the eigenvalue is real, the associated row of the complex matrix (say G_{zs}) is transferred to a real matrix (say $G_{zs_{real}}$). If the ith eigenvalue is the first or second of a complex conjugate pair, the real or complex part of the ith row of G_{zs} is transferred to the ith or i + 1st row of $G_{zs_{real}}$, respectively. For example, if (in complex notation)

$$EIG = [\ (1,\ 0)\quad (1,\ 1)\quad (1,\ -1)\quad (2,\ 0)\ ] \tag{60}$$

and

$$G_{zs} = \begin{bmatrix} (1,\ 0) & (2,\ 0) & (3,\ 0) & (4,\ 0) \\ (2,\ 1) & (1,\ 3) & (3,\ -2) & (5,-4) \\ (2,-1) & (1,-3) & (3,\ 2) & (5,\ 4) \\ (3,\ 0) & (2,\ 0) & (5,\ 0) & (1,\ 0) \end{bmatrix} \tag{61}$$

then

$$G_{zs_{real}} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 3 & 5 \\ 1 & 3 & -2 & -4 \\ 3 & 2 & 5 & 1 \end{bmatrix} \tag{62}$$

This procedure is possible since IMSL subroutine EIGRF keeps complex conjugate pairs of eigenvalues and eigenvectors together (Ref 5: EIGRF).

The singular values and condition number of G or G_{zs} are then printed. If at this time the user wishes to change the system matrices and vectors, he must type "^A" (return) and begin the program again. If no change is desired, the program forms the time- and input-dependent parts of the sensitivity matrix.

Time- and Input-Dependent Parts of S. Once G has been formed, the program prompts the user to input the sample spacing Δ, the number of samples taken K, and the discretized input U_d at each time t_k.

The matrices E and F are formed according to Eqs (38) through (45); and if $\underline{x}_o$ is not the zero vector, S is computed using Eq (33). If $\underline{x}_o$ is the zero vector, then

$$S = F * G_{zs} \tag{63}$$

since G_{zi} and, therefore, $E * G_{zi}$ have all zero elements. The user can perform a singular value decomposition on S, if desired. Since S is a real matrix (for the real matrix A) but has complex notation in the program, the real matrix S_{real} simply contains the real parts of the complex matrix S. IMSL subroutine LSVDF may then be called to perform a singular value decomposition of S, and the condition number of S may be calculated.

Computational Load

With the assumptions of constant sample spacing Δ and a piecewise constant input U, the computations of matrices E and F become extremely efficient. For details, see the development of E and F in Appendix A (Resulting equations are presented in Chapter II.) and subroutine EFMAT in the program listed in Appendix B.

Subroutine GMATR, in which both G_{zi} and G_{zs} are formed, involves only the multiplications and additions shown in the second half of Eq (47). Subroutine SENS calculates the sensitivities needed for forming G, as explained earlier.

The full computational load is presented in Table I. As shown in the table, the number of multiplications needed to compute G is of order NA * NPA, and the number of multiplications needed to form E and F is of order NA * K. Thus with the assumptions of constant sample spacing, piecewise constant input, and the linear appearance of parameters, the computational load of this algorithm is small.

Variations of the Assumptions

If the assumptions on the sample spacing, input and parameters are changed, implementing the algorithm becomes more difficult. The multi-input multi-output case involves more computation but does not increase the complexity of the computations. Examples are presented in the following sections.

Unequal Sample Spacing. If the sample spacing Δ is allowed to vary, then the recursive formulas of the E matrix are no longer valid. For example, $e^{\lambda_\ell t_2}$ does not necessarily equal $e^{\lambda_\ell (2\Delta)}$ (See Eq (38).), and $e^{\lambda_\ell t_k}$ must be calculated for each ℓ = 1, 2, . . . , NP and k = 1, 2, . . . , K. New formulas must also be derived to compute the F matrix.

TABLE I

COMPUTATIONAL LOAD OF MODAL ALGORITHM

Code Section	Number of Multiplications		Number of Additions	
IMSL: EIGRF (EIGCC)	NA^3		(1)	
IMSL: LEQT1C	NA^2	(2)	NA^2	(2)
IMSL: LSVDF	$2K(NP)^2$	(3)	$2K(NP)^2$	(3)
Vectors independent of parameters	3NA		3NA	
Sensitivities of eigenvalues $\lambda_{j(i)}$	NA * NPA		0	
Sensitivities of eigenvectors $\underline{u}_{j(i)}$ and $\underline{v}_{j(i)}$	2NA * NPA		NA * NPA	
Sensitivities in $\underline{x}_o$, $\underline{B}$ and $\underline{C}$	0		0	
Calculation of G_{zi} and G_{zs}	2NA * (NP + 1)		2NA * NP	
Calculation of E and F	9NA * (K + 1)		3NA * (K + 1)	
Calculation of S				
$G_{zi} \neq 0$	4K * NA * NP		3K * NP	
$G_{zi} = 0$	2K * NA * NP		K * NP	
Calculation of $\frac{1}{\kappa}$	1		0	

Notes: (1) Values unavailable

(2) Ref 3: 1.22

(3) Ref 3: 11.18

Input Not Piecewise Constant. If the input is not piecewise constant, the integrals of the F matrix must be integrated using numerical analysis methods. The equations for computing F become more complex, and their calculation requires more computer time.

Nonlinear Appearance of Parameters. If the parameters are allowed to appear nonlinearly in the vectors $\underline{B}$, $\underline{C}$ and $\underline{x}_0$, the sensitivities of the appropriate terms of these vectors would not necessarily be unity. Therefore, the partial derivatives with respect to the parameters would have to be taken and multiplied to the corresponding elements of $\underline{u}_j$ and $\underline{v}_j'$. The values would have to be placed in the appropriate locations of $\underline{C}_{(i)}\ \underline{u}_j$, $\underline{v}_j'\ \underline{x}_{o(i)}$ and $\underline{v}_j'\ \underline{B}_{(i)}$. For example, if

$$\underline{C} = [\ 2 \quad \theta_5^2\ ] \tag{64}$$

where $\theta_5 = 2$, and

$$\underline{u}_j = [\ 3 \quad 4\ ]' \tag{65}$$

then

$$\underline{C}_{(5)}\ \underline{u}_j = [\ 0 \quad 2\theta_5] * \begin{bmatrix} 3 \\ 4 \end{bmatrix} = 16 \tag{66}$$

Similarly, if θ_i appears nonlinearly in the (ℓ, m) location of A, then $a_{\ell, m(i)}$ must be calculated and multiplied to the right-hand sides of Eqs (53), (58) and (59).

The Multi-Input Multi-Output Case. For a single-input single-output system the vectors $\underline{B}$ and $\underline{C}$ of Eqs (1) and (3) have dimension NA x 1 and 1 x NA, respectively. However, if NB equals the number of inputs and NC equals the number of outputs, $\underline{B}$ and $\underline{C}$ become matrices B and C of dimension NA x NB and NC x NA, respectively. Then the program LKP would become a subroutine to be called NB * NC times from a new main program. For example,

if the system is described by the equations

$$\underline{\dot{x}} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \underline{x} + \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} \tag{67}$$

$$\underline{y}(t_k) = \begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix} \underline{x}(t_k) \tag{68}$$

then one of the four calls to the "subroutine" LKP would send in the information

$$\underline{\dot{x}} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \underline{x} + \begin{bmatrix} 1 \\ 3 \end{bmatrix} U_1 \tag{69}$$

$$y_2(t_k) = [\ 4 \quad 5\ ]\ \underline{x}(t_k) \tag{70}$$

The "subroutine" LKP would compute four sensitivity matrices, one for each input-output pair.

Summary

The use of the modal algorithm and the computer program implementing the algorithm in the experimental design plase and in the actual estimation task has been noted. The computational efficiencies of the program have also been summarized, and the computational load of the program has been tabulated. In addition, the effects of changing some of the basic assumptions of the algorithm have been mentioned and exemplified. In the next chapter, the algorithm will be verified and used to investigate several experimental design issues.

IV. Results

Not only can the sensitivity matrix be calculated using the new algorithm developed in Chapter II, but several experimental design issues can be addressed. The design issues are divided into two categories: input design issues and structural design issues. Under the category of input design issues fall changes in the sample spacing Δ, the number of sample times K, and the input $U(t_k)$. Structural design issues include choosing a zero or nonzero initial state and varying the form of the mathematical model by adding zeros or poles to the transfer function or by choosing different canonical forms to represent the system. Since the program implementing the algorithm is interactive, the user may easily change the input and structural design to find the best model of the system before the final experiment must be performed.

In the first section of this chapter, the algorithm is verified by comparing its result with that of the "standard" sensitivity system method. In the second section, the experimental design issues of several examples are investigated.

Verification of the Algorithm

To verify the algorithm, the sensitivity matrices of two systems are formed by using the sensitivity system method and then by using the new algorithm.

The first system is described by the equations

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 1 \\ -2 & -2 \end{bmatrix} \underline{x} + \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} U \tag{71}$$

$$y(t_k) = [\theta_3 \quad \theta_4]\ \underline{x}(t_k) \tag{72}$$

where $\underline{\theta} = [0 \quad 1 \quad 10 \quad 100]'$. The sensitivity system results in a set of six (NA * (NPA + NPB + NPC) + 1 = 6) state and input equations and five (NP + 1 = 5) output equations to be integrated from 0 to 1 with $\Delta = 0.1$:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_{1(1)} \\ \dot{x}_{2(1)} \\ \dot{x}_{1(2)} \\ \dot{x}_{2(2)} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -2 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_{1(1)} \\ x_{2(1)} \\ x_{1(2)} \\ x_{2(2)} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} U \tag{73}$$

$$\begin{bmatrix} y(t_k) \\ y_{(1)}(t_k) \\ y_{(2)}(t_k) \\ y_{(3)}(t_k) \\ y_{(4)}(t_k) \end{bmatrix} = \begin{bmatrix} 10 & 100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 100 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 100 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 10 & 100 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 100 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_{1(1)} \\ x_{2(1)} \\ x_{1(2)} \\ x_{2(2)} \\ x_{1(3)} \\ x_{2(3)} \\ x_{1(4)} \\ x_{2(4)} \end{bmatrix} \tag{74}$$

Three cases are considered:

1. Case I: $\underline{x}_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $U_d(t_k) = 0$ for $k = 1, 2, \ldots, 10$, (zero input).

2. Case II: $\underset{\sim}{x}_o = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $U_d(t_k) = 0$ for $k = 1, 2, \ldots, 10$ (zero input).

3. Case III: $\underset{\sim}{x}_o = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $U_d(t_k) = 1$ for $k = 1, 2, \ldots, 10$ (zero state, unit step input).

The ODE software package (Ref 10) was used for the integration. The sensitivity matrices formed by both methods may be seen in Tables II, III and IV. In Cases I and II, the input corresponding to the first two parameters (both located in the input vector $\underline{B}$) is zero, with the result that the first two columns of the sensitivity matrices are identically zero. Therefore, only the second two columns appear in Tables II and III.

As seen in Tables II through IV, the numerical results of both methods are very close. The reader might note that the new algorithm is inherently more accurate than the sensitivity system calculation since the errors of numerical integration are not introduced in the modal sensitivity method. Unfortunately, as the eigenvalues of A approach each other, the sensitivity matrix tends to become ill-conditioned for both methods. The output response, however, may be relatively insensitive to changes in the parameters even when the parallel eigenvectors drive the sensitivity matrix to infinity. To avoid this situation, the system may be written in a block diagonal form. Then the eigenvalues of A lie along the main diagonal of A, and the eigenvector and reciprocal eigenvector sensitivities are identically zero. This not only simplifies the formation of the sensitivity matrix, but also alleviates the problem of nearly equal eigenvalues. This problem and its solution are discussed in detail by Reid and Palmer (Ref 16: 21-33).

Another system used to verify the modal algorithm is described by the

TABLE II

VERIFICATION: CASE I

Sensitivity of $y(t_k)$ with Respect to θ_i

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 1 \\ -2 & -2 \end{bmatrix} \underline{x} + \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} U \qquad \text{Zero Input} \quad \underline{x}_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$y(t_k) = \begin{bmatrix} \theta_3 & \theta_4 \end{bmatrix} \underline{x}(t_k) \qquad \underline{\theta} = \begin{bmatrix} 0 & 1 & 10 & 100 \end{bmatrix}'$$

	Sensitivity System Method		Modal Method	
t_k	$y_{(3)}(t_k)$	$y_{(4)}(t_k)$	$y_{(3)}(t_k)$	$y_{(4)}(t_k)$
0.1	.9906500107947	-.1806660218957	.9906500107976	-.1806660219048
0.2	.9650673381709	-.3253133816401	.9650673381578	-.3253133816307
0.3	.92665743169	-.4378535073505	.9266574317007	-.4378535073487
0.4	.8784405689912	-.5220698422676	.8784405690451	-.5220698422869
0.5	.823067018328	-.5815725763732	.8230670184283	-.5815725764254
0.6	.7628361486861	-.6197647192039	.7628361487773	-.6197647192642
0.7	.6997184257697	-.6398180717845	.6997184258498	-.6398180718494
0.8	.6353793731481	-.6446577383981	.6353793732315	-.6446577384541
0.9	.5712047083421	-.6369539102161	.571204708404	-.6369539102258

TABLE III

VERIFICATION: CASE II

Sensitivity of $y(t_k)$ with Respect to θ_i

$$\underline{\dot{x}} = \begin{bmatrix} 0 & 1 \\ -2 & -2 \end{bmatrix} \underline{x} + \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} U \qquad \text{Zero Input} \quad \underline{x}_0 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$y(t_k) = \begin{bmatrix} \theta_3 & \theta_4 \end{bmatrix} \underline{x}(t_k) \qquad \underline{\theta} = \begin{bmatrix} 0 & 1 & 10 & 100 \end{bmatrix}'$$

	Sensitivity System Method		Modal Method	
t_k	$y_{(3)}(t_k)$	$y_{(4)}(t_k)$	$y_{(3)}(t_k)$	$y_{(4)}(t_k)$
0.1	.09033301093665	.8099839889108	.09033301095242	.8099839888928
0.2	.1626566908265	.6397539565384	.1626566908153	.6397539565272
0.3	.218926753687	.4888039243323	.2189267536743	.488803924352
0.4	.2610349211446	.3563707267115	.2610349211434	.3563707267582
0.5	.2907862881656	.2414944418882	.2907862882127	.241494442003
0.6	.2098823595139	.143071429361	.3098823596321	.1430714295131
0.7	.3199090357938	.05990035391247	.3199090359247	.05990035400043
0.8	.3223288691002	.009278365277813	.322328869227	-.009278365222578
0.9	.3184769549788	-.065749201844484	.3184769551129	-.06574920182172

TABLE IV

VERIFICATION: CASE III

Sensitivity of $y(t_k)$ with Respect to 0_i

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 1 \\ -2 & -2 \end{bmatrix} \underline{x} + \begin{bmatrix} 0_1 \\ 0_2 \end{bmatrix} u \qquad \text{Unit Step Input} \quad \underline{x}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$y(t_k) = \begin{bmatrix} 0_3 & 0_4 \end{bmatrix} \underline{x}(t_k) \qquad \underline{0} = \begin{bmatrix} 0 & 1 & 10 & 100 \end{bmatrix}'$$

	Sensitivity System Method				Modal Method			
t_k	$y_{(1)}(t_k)$	$y_{(2)}(t_k)$	$y_{(3)}(t_k)$	$y_{(4)}(t_k)$	$y_{(1)}(t_k)$	$y_{(2)}(t_k)$	$y_{(3)}(t_k)$	$y_{(4)}(t_k)$
0.1	.061831080876	9.08005104059	.004674994603	.090333010945	.061831081309	9.08005104125	.0046749946011	.09033301095242
0.2	-1.5173726554	16.4403323915	.107466330909	.162656690824	-1.5173726576	16.4403323907	.0174663309210	.1626566908153
0.3	-4.4115636097	22.2593882095	.036671284147	.218926753680	-4.4115636101	22.2593882089	.0366712841496	.2189267536743
0.4	-8.3299995793	26.7112892686	.060779715503	.261034921136	-8.3299995745	26.7112892691	.0607797154774	.2610349211434
0.5	-130161054717	29.9632937264	.088466490853	.290786288179	-13.016105459	29.9632937291	.0884664907858	.29078628882127
0.6	-18.245923029	32.1740552148	.118581925693	.309882359579	-18.245923013	32.1740552193	.1185819256113	.3098828596321
0.7	-23.826251328	33.4923114587	.150140787149	.319909035872	-23.826251314	33.4923114632	.1501407870751	.3199090359247
0.8	-29.592567729	34.0559900512	.182310313453	.322328869166	-29.592567716	34.0559900565	.1823103133842	.322328869227
0.9	-35.406806703	33.9916719684	.214397645857	.318476955098	-35.406806692	33.9916719692	.214397645798	.3184769551129

equations

$$\dot{\underline{x}} = \begin{bmatrix} \theta_1 & 0 \\ 0 & \theta_2 \end{bmatrix} \underline{x} + \begin{bmatrix} \theta_3 \\ \theta_4 \end{bmatrix} U \tag{75}$$

$$y(t_k) = \begin{bmatrix} \theta_5 & \theta_6 \end{bmatrix} \underline{x}(t_k) \tag{76}$$

When the nominal values of the parameters are $\underline{\theta} = \begin{bmatrix} -1 & -2 & 1 & 1 & 10 & 100 \end{bmatrix}'$,

the system becomes

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_{1(1)} \\ \dot{x}_{2(1)} \\ \dot{x}_{1(2)} \\ \dot{x}_{2(2)} \\ \dot{x}_{1(3)} \\ \dot{x}_{2(3)} \\ \dot{x}_{1(4)} \\ \dot{x}_{2(4)} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_{1(1)} \\ x_{2(1)} \\ x_{1(2)} \\ x_{2(2)} \\ x_{1(3)} \\ x_{2(3)} \\ x_{1(4)} \\ x_{2(4)} \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} U \tag{77}$$

and

$$\begin{bmatrix} y(t_k) \\ y_{(1)}(t_k) \\ y_{(2)}(t_k) \\ y_{(3)}(t_k) \\ y_{(4)}(t_k) \\ y_{(5)}(t_k) \\ y_{(6)}(t_k) \end{bmatrix} = \begin{bmatrix} 10 & 100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 100 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 10 & 100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 100 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_{1(1)} \\ x_{2(1)} \\ x_{1(2)} \\ x_{2(2)} \\ x_{1(3)} \\ x_{2(3)} \\ x_{1(4)} \\ x_{2(4)} \\ x_{1(5)} \\ x_{2(5)} \\ x_{1(6)} \\ x_{2(6)} \end{bmatrix} \qquad (78)$$

The zero state, unit step case sensitivity matrices formed by the two methods appear in Table V.

Design Issues

To investigate structural design issues, the system transfer function

$$G(s) = \frac{2}{(s + 1)\ (s + 2)} = \frac{2}{s^2 + 3s + 2} \qquad (79)$$

is expressed in both the phase variable and the diagonal canonical forms, the zero state part of the G matrix (G_{zs}) is formed for both cases, and their condition numbers are calculated and compared. By comparing the

TABLE V

VERIFICATION: CASE IV

Sensitivity of $y(t_k)$ with Respect to θ_i

$$\dot{\underline{x}} = \begin{bmatrix} \theta_1 & 0 \\ 0 & \theta_2 \end{bmatrix} \underline{x} + \begin{bmatrix} \theta_3 \\ \theta_4 \end{bmatrix} U \qquad \text{Zero Initial State, Unit Step Input}$$

$$y(t_k) = \begin{bmatrix} \theta_5 & \theta_6 \end{bmatrix} \underline{x}(t_k) \qquad \underline{\theta} = \begin{bmatrix} -1 & -2 & 1 & 1 & 10 & 100 \end{bmatrix}'$$

Sensitivity System Method

t_k	$y_{(1)}(t_k)$	$y_{(2)}(t_k)$	$y_{(3)}(t_k)$	$y_{(4)}(t_k)$	$y_{(5)}(t_k)$	$y_{(6)}(t_k)$
0.1	0.0467884015774	0.4380774052206	0.951625819646	9.063462347321	0.09516258196462	0.09063462347321
0.2	0.1752309630962	1.538798401962	1.812692469216	16.48399769392	0.1812692469216	0.1648399769393
0.3	0.3693631311641	3.047534566038	2.591817793179	22.55941819201	0.2591817793179	0.2255941819201
0.4	0.6155193555299	4.780196626955	3.29679953964	27.53355179023	0.329679953964	0.2753355179023
0.5	0.9020401043433	6.606027957135	3.934693402869	31.60602793658	0.3934693402869	0.3160602793658
0.6	1.219013822532	8.434318362814	4.511883639055	34.94028939887	0.451883639055	0.3494028939887
0.7	1.558049835578	10.20418217722	5.034146962081	37.67015179854	0.5034146962081	0.3767015179854
0.8	1.912078645917	11.87672634009	5.506710358824	39.905174097	0.5506710358824	0.39905174097
0.9	2.275176464958	13.42907784357	5.93430340249	41.73505558359	0.593430340259	0.4173505558359

TABLE V (cont.)

VERIFICATION: CASE IV (cont.)

Sensitivity of $y(t_k)$ with Respect to θ_i

$$\underline{\dot{x}} = \begin{bmatrix} \theta_1 & 0 \\ 0 & \theta_2 \end{bmatrix} \underline{x} + \begin{bmatrix} \theta_3 \\ \theta_4 \end{bmatrix} U \qquad \text{Zero Initial State, Unit Step Input}$$

$$y(t_k) = \begin{bmatrix} \theta_5 & \theta_6 \end{bmatrix} \underline{x}(t_k) \qquad \underline{\theta} = \begin{bmatrix} -1 & -2 & 1 & 1 & 10 & 100 \end{bmatrix}'$$

	Modal Method					
t_k	$y_{(1)}(t_k)$	$y_{(2)}(t_k)$	$y_{(3)}(t_k)$	$y_{(4)}(t_k)$	$y_{(5)}(t_k)$	$y_{(6)}(t_k)$
0.1	0.4678840160448	0.4380770407660	0.9516258196404	9.063462346101	0.09516258196404	0.9063462346101
0.2	0.1752309630643	1.538798388753	1.81269246922	16.48399769822	0.181269246922	0.1648399769822
0.3	0.3693631311378	3.047534556239	2.591817793183	22.5594181953	0.2591817793183	0.225594181953
0.4	0.6155193555012	4.780196614726	3.296799539644	27.53355179414	0.3296799529644	0.2753355179414
0.5	0.9020401043108	6.606027941429	3.934693402874	31.60602794143	0.3934693402874	0.3160603794143
0.6	1.219013822496	8.43431834483	4.51188363906	34.94028940439	0.451188363906	0.3494028940439
0.7	1.558049835546	10.2041821635	5.034146962086	37.67015180292	0.5034146962086	0.3767015180292
0.8	1.91207864589	11.87672633035	5.506710358828	39.90517410027	0.5506710358828	0.3990517410027
0.9	2.275176464929	13.42907782449	5.934303402594	41.73505558892	0.5934303402594	0.4173505558892

condition numbers, a conclusion may be drawn as to which form is better conditioned and, thus, preferable over the other for representing the system in the identification/estimation problem. Then one of the forms is chosen to investigate two of the input design issues mentioned earlier. By calculating and plotting the inverse condition numbers of successive trials, the effects on the output sensitivity of changing Δ and K may be noted.

<u>Structural Design Issues</u>. The phase variable form of Eq (79) is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \theta_1 & \theta_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{80}$$

$$y(t_k) = \begin{bmatrix} \theta_3 & \theta_4 \end{bmatrix} \underline{x}(t_k) \tag{81}$$

where $\underline{\theta} = \begin{bmatrix} -2 & -3 & 2 & 0 \end{bmatrix}'$.

The diagonal form is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \theta_1 & 0 \\ 0 & \theta_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u \tag{82}$$

$$y(t_k) = \begin{bmatrix} \theta_3 & \theta_4 \end{bmatrix} \underline{x}(t_k) \tag{83}$$

where $\underline{\theta} = \begin{bmatrix} -1 & -2 & 2 & -2 \end{bmatrix}'$.

The input is of no interest for the purposes of this investigation; therefore, only the inverse of the condition number $\frac{1}{\kappa}$ is recorded. As shown in Table VI, the diagonal form has a better condition number than the phase variable form has. This result is typical, implying that the

TABLE VI

VALUES FOR INVERSE CONDITION NUMBER OF

STRUCTURE-DEPENDENT PART OF SENSITIVITY MATRIX

$$G(s) = \frac{2}{s^2 + 3s + 2} \qquad K = 10, \quad \Lambda = 0.1, \qquad \text{Zero initial state}$$

Unit step input

Zero at	Phase Variable Form	Diagonal Form
(1)	0.0295	0.5000
-2.005	0.0007	0.0050
-10.010	0.0070	0.1111

Note: (1) No zeros in transfer function.

output sensitivity matrix using the diagonal form remains better conditioned as model parameters change than does that using the phase variable form. The reason for its better conditioning is that the diagonal form gives a balance of the sensitivity among the various parameters, yielding a better conditioning of the inherent linear equation problem of quasilinearization. On the other hand, the phase variable form may give a high sensitivity to some parameters and low sensitivity to others. This imbalance in the sensitivity yields a poorly conditioned problem which has large errors in certain "directions," as discussed in Chapter II.

To further compare the canonical forms and to see the effect of adding zeros to the transfer function of a system, a zero is added at -2.005 (close to the pole at -2) and than at -10.01 (far away from both poles). In both instances, the condition number of the diagonal form is better than that of the phase variable form, indicating that the diagonal form is superior in estimating parameters. With the addition of zeros, the condition of both forms becomes worse (See Table VI.), indicating that the addition of zeros to the transfer function makes parameter estimation more difficult. Another revelation of Table VI is that placing a zero over a pole may lessen the accuracy of the parameter estimation enough to make it totally unreliable.

Input Design Issues. The investigation of input design issues includes observing how changing sample spacing Δ, number of samples taken K, or the input $U_d(t_k)$ affects the output. In this section the phase variable form of Eq (79) is chosen since its condition number varies more than that of the diagonal form, as may be se[illegible]ole VI. Therefore, any correlation between the condition number and changes in Δ or K should be more apparent with this form.

Figure 1 shows how $\frac{1}{\kappa}$ varies with when 10 samples are taken for the zero state, unit step input case. With K held constant, the condition number improves ($\frac{1}{\kappa}$ approaches 1) as the sample spacing is increased, and then deteriorates when the sample spacing becomes too large. The first trend may be surprising, but it only means that the whole interval $K * \Delta$ over which the sample outputs are taken becomes larger; and more information is gained. As the interval becomes too large, however, not enough information is taken between sample times to learn the true output response. This is reflected by the decrease in $\frac{1}{\kappa}$.

Out of Figure 1 comes the unanticipated discovery that for each number of sample times, there is an optimum sample spacing (and, therefore, an optimum interval) over which to observe the output response. In the following figures, other systems are examined to find the optimum sample spacing for a given number of samples.

Figure 2 shows the results for the system

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 1 \\ \theta_1 & \theta_2 \end{bmatrix} \underline{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} U \tag{84}$$

$$y(t_k) = \begin{bmatrix} \theta_3 & \theta_4 \end{bmatrix} \underline{x}(t_k) \tag{85}$$

where $\underline{\theta} = \begin{bmatrix} -0.01 & -2 & 0.01 & 0 \end{bmatrix}'$ and where 10 samples are taken with a unit step input. The results for the system of Eqs (80) and (81) with $U = \sin t$ is shown in Figure 3.

Figure 4 shows the relationship of the inverse condition number with sample spacing for the system

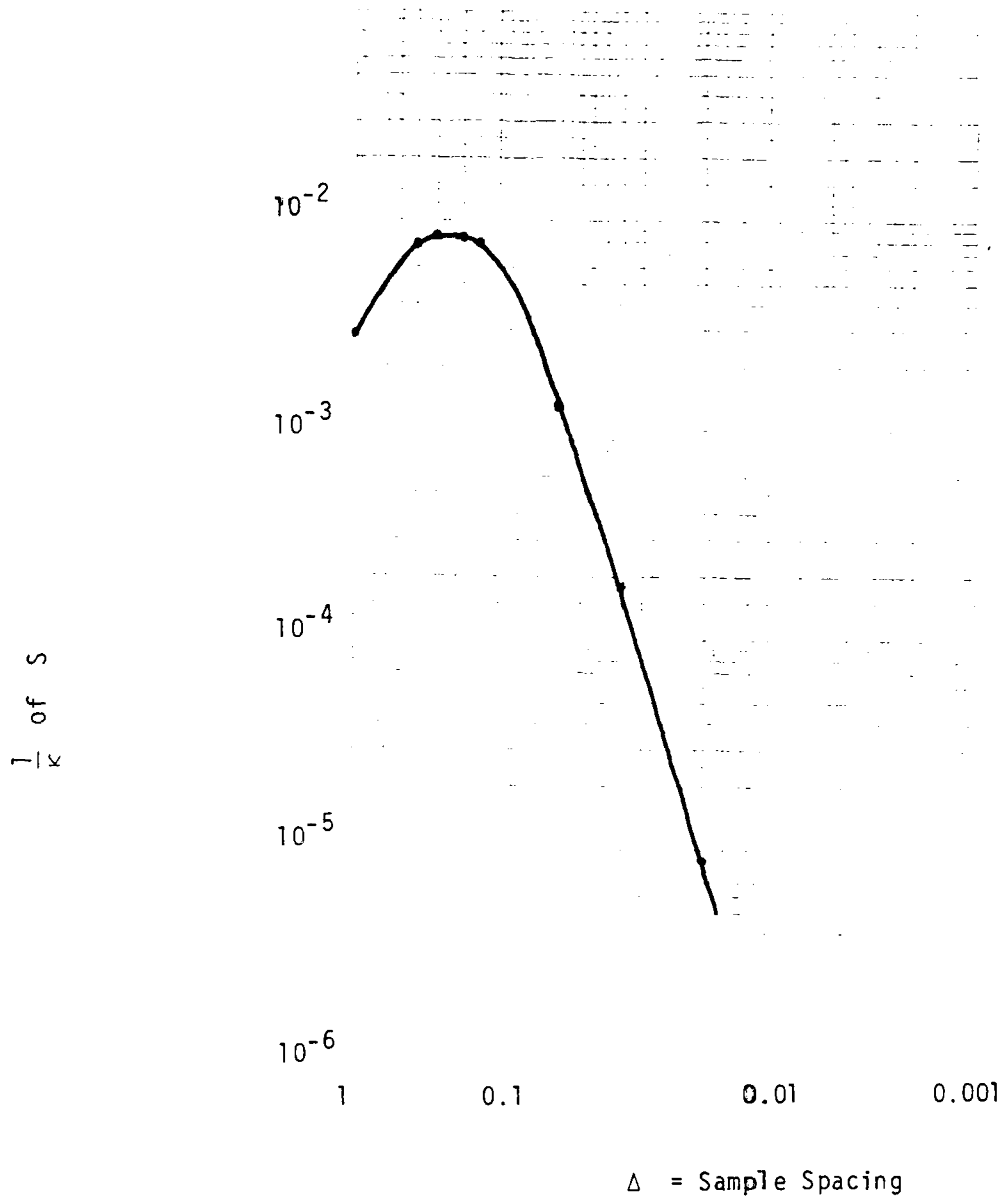

Figure 1. Inverse Condition Number vs. Sample Spacing for System of Eqs (80) and (81) with K = 10 and Unit Step Input

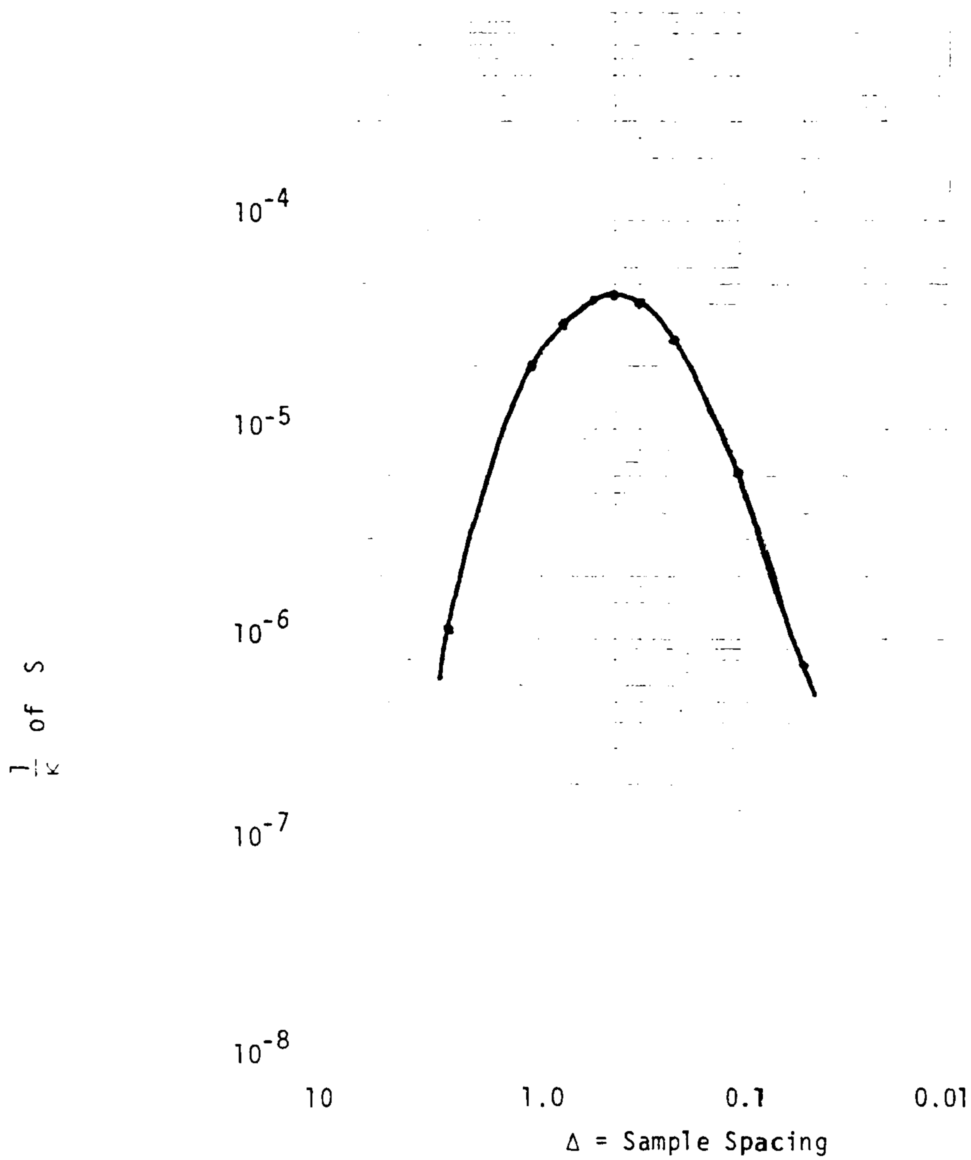

Figure 2. Inverse Condition Number vs. Sample Spacing for System of Eqs (84) and (85) with K = 10 and Unit Step Input

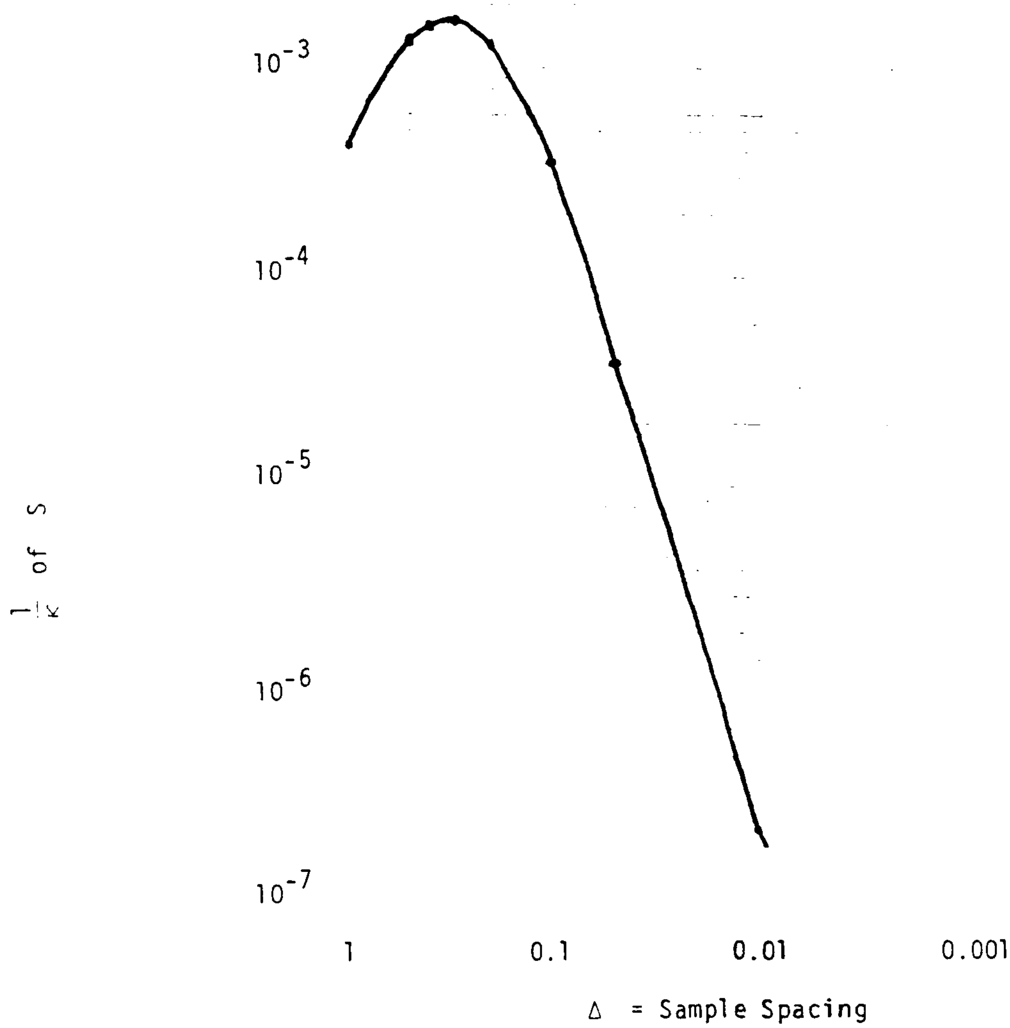

Figure 3. Inverse Condition Number vs. Sample Spacing for System of Eqs (80) and (81) with K = 10 and U = sin t

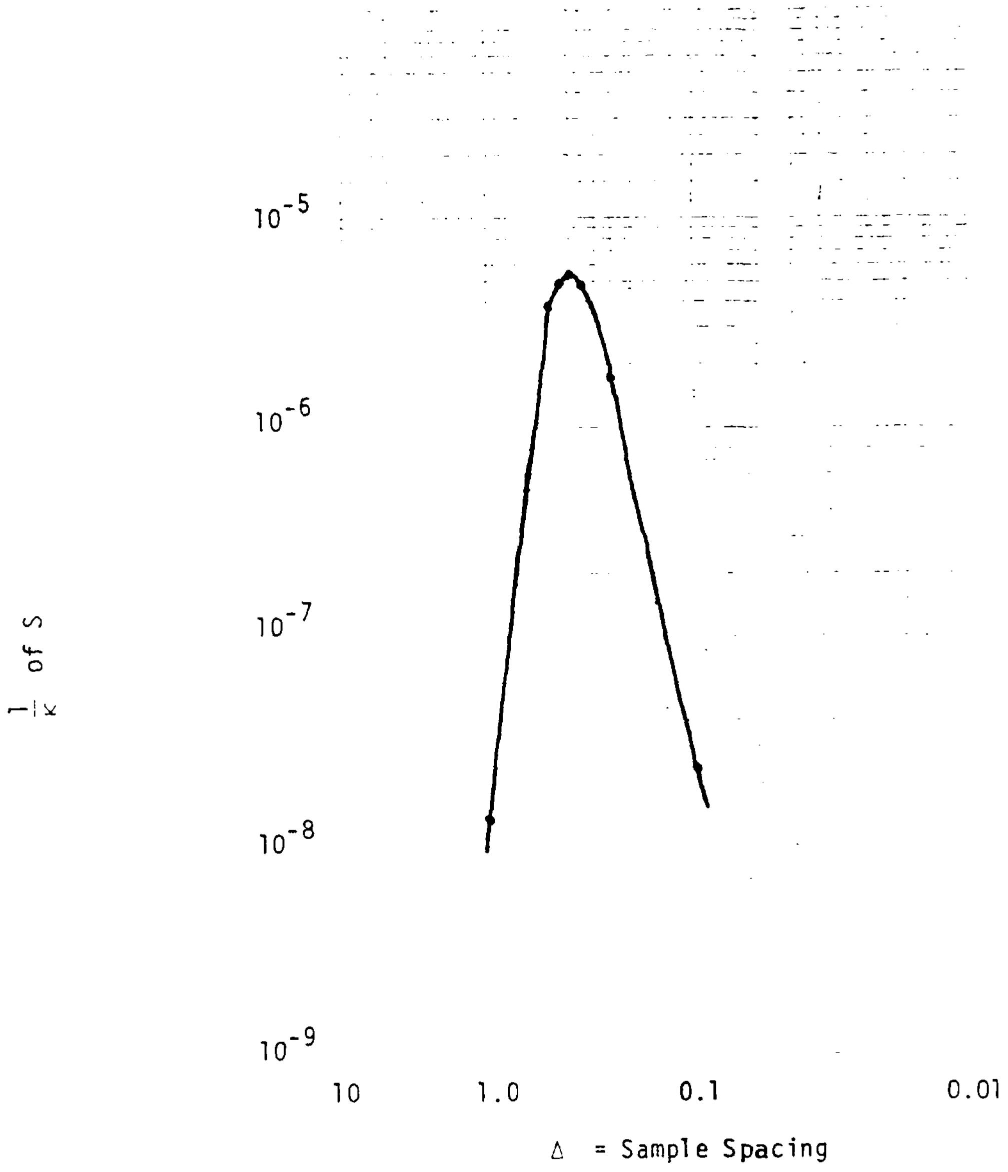

Figure 4. Inverse Condition Number vs. Sample Spacing for System of Eqs (86) and (87) with K = 10 and Unit Step Input

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \theta_1 & \theta_2 & \theta_3 & \theta_4 \end{bmatrix} \underline{x} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 100 \end{bmatrix} U \tag{86}$$

$$y(t_k) = \begin{bmatrix} \theta_5 & \theta_6 & \theta_7 & \theta_8 \end{bmatrix} \underline{x}(t_k) \tag{87}$$

where $\underline{\theta} = \begin{bmatrix} -5.05 & -7.06 & -7.21 & -6.2 & 0.055 & 1.15 & 1.00 & 0.00 \end{bmatrix}'$ with a unit step input, zero initial state and 10 samples taken. The curve for the fourth order system is steeper around the optimum point than previous curves. Therefore, it seems that determining the optimum sample spacing during the design phase is more important as the system becomes larger (i. e., as the number of parameters increases).

The relationship of the inverse condition number with the number of samples taken when the sample spacing is held constant for the system of Eqs (80) and (81) is shown in Figure 5. A unit step input is used. As expected, for constant $\Delta = 0.1$, an increase in the number of samples taken results in a better condition number of the sensitivity matrix until a point is reached after which taking more samples does little good.

The results for the same system where $U = \sin t$ may be seen in Figure 6. Figure 7 shows the results for the system of Eqs (84) and (85) with a unit step input and $\Delta = 0.1$.

In Figure 8, the relationship between the inverse condition number of S and the number of samples taken for three different sample spacings is shown. Using this type of graph, the designer may choose an optimum combination of sample spacing and number of samples. As in Figure 8, the

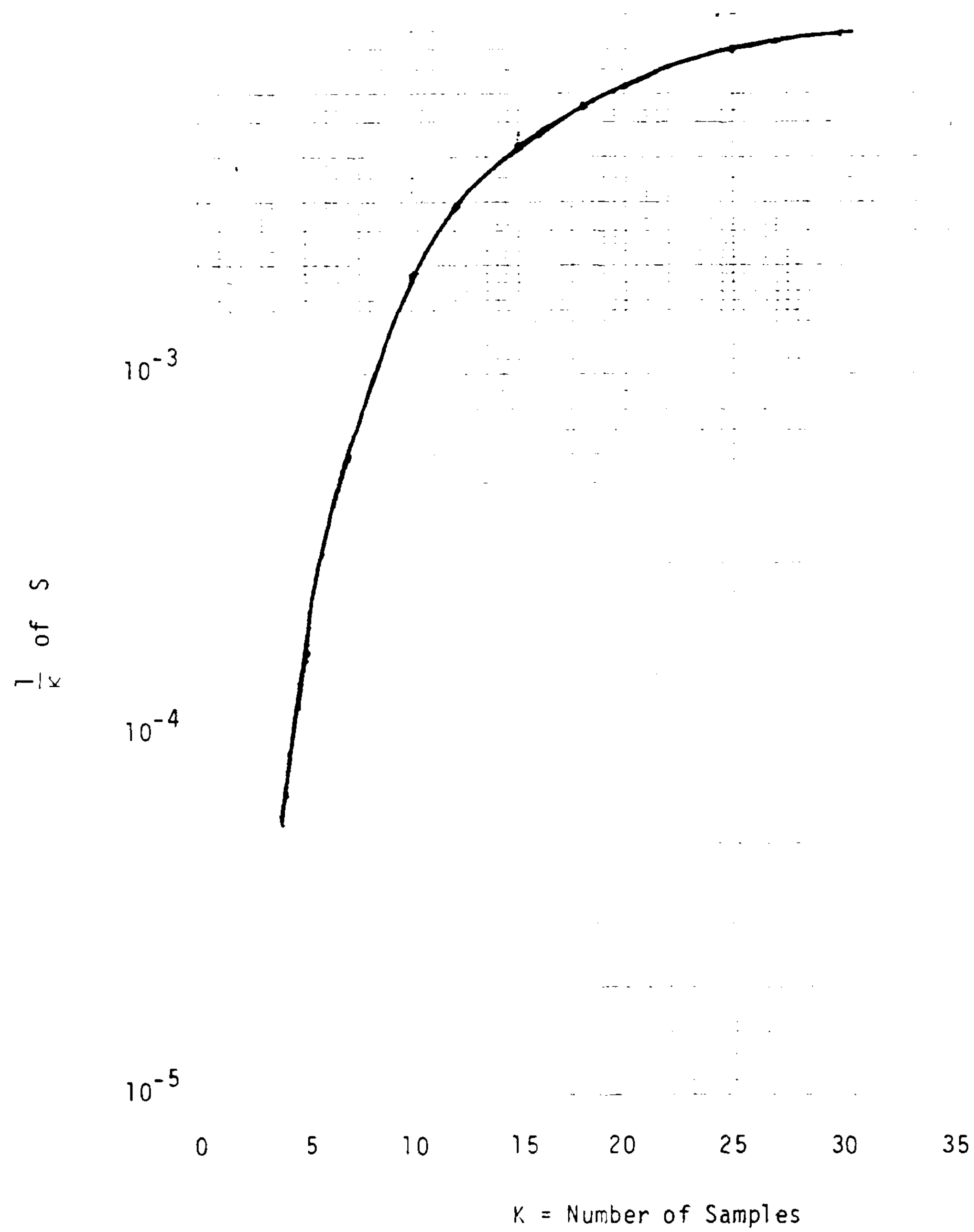

Figure 5. Inverse Condition Number vs. Number of Samples Taken for System of Eqs (80) and (81) with $\Delta = 0.1$ and Unit Step Input

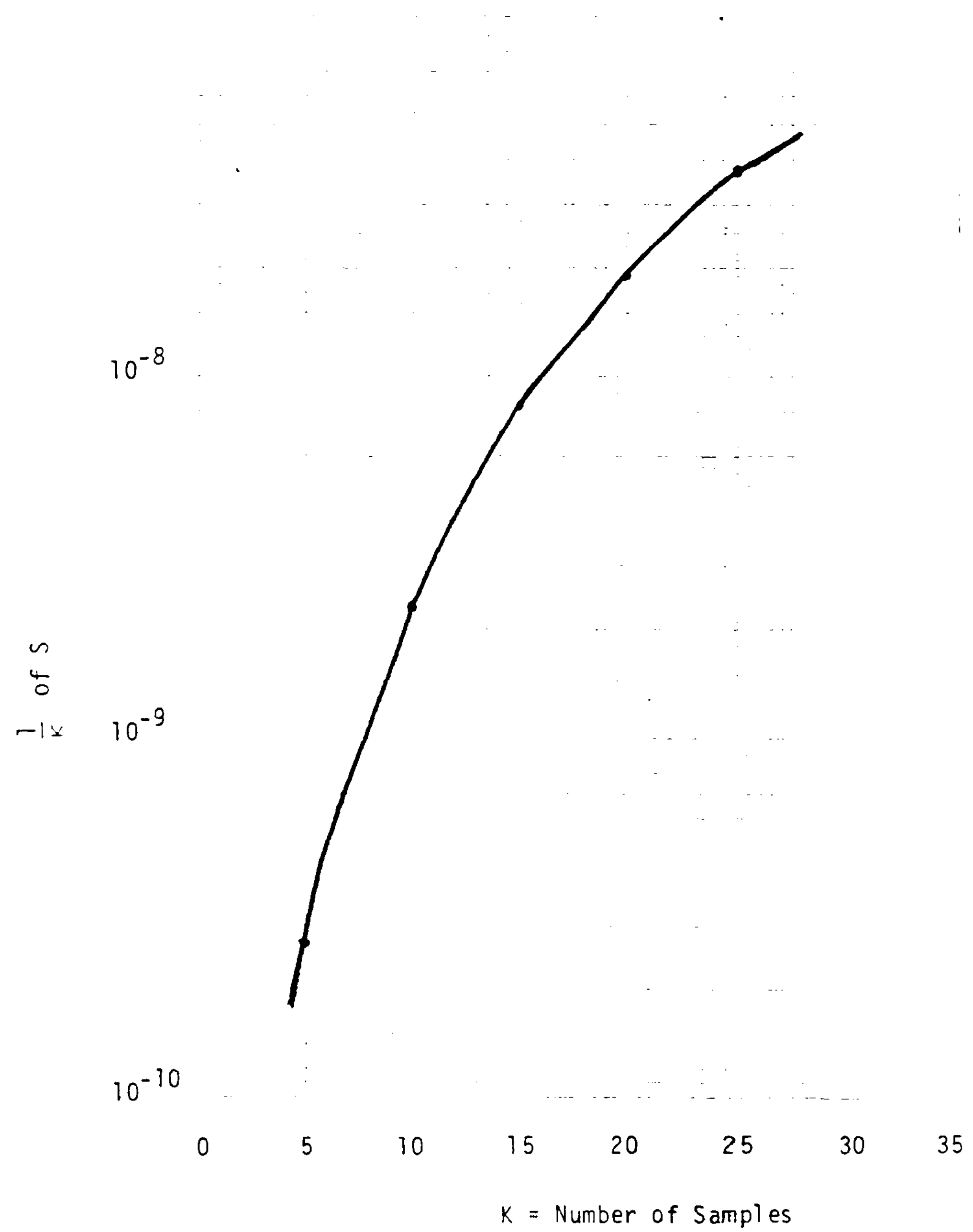

Figure 6. Inverse Condition Number vs. Number of Samples Taken for System of Eqs (80) and (81) with $\Delta = 0.1$ and $U = \sin t$

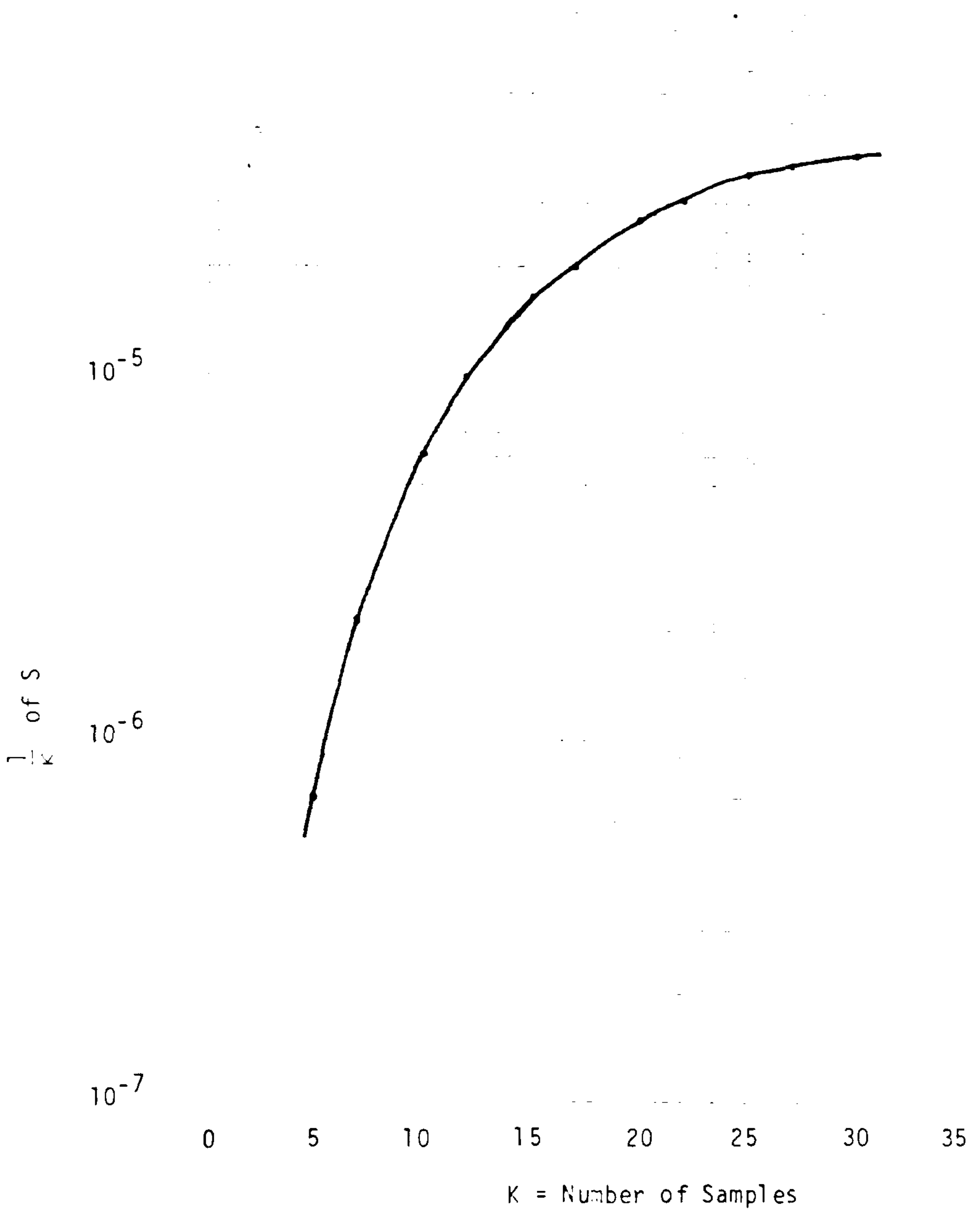

Figure 7. Inverse Condition Number vs. Number of Samples Taken for System of Eqs (84) and (85) with Δ = 0.1 and Unit Step Input

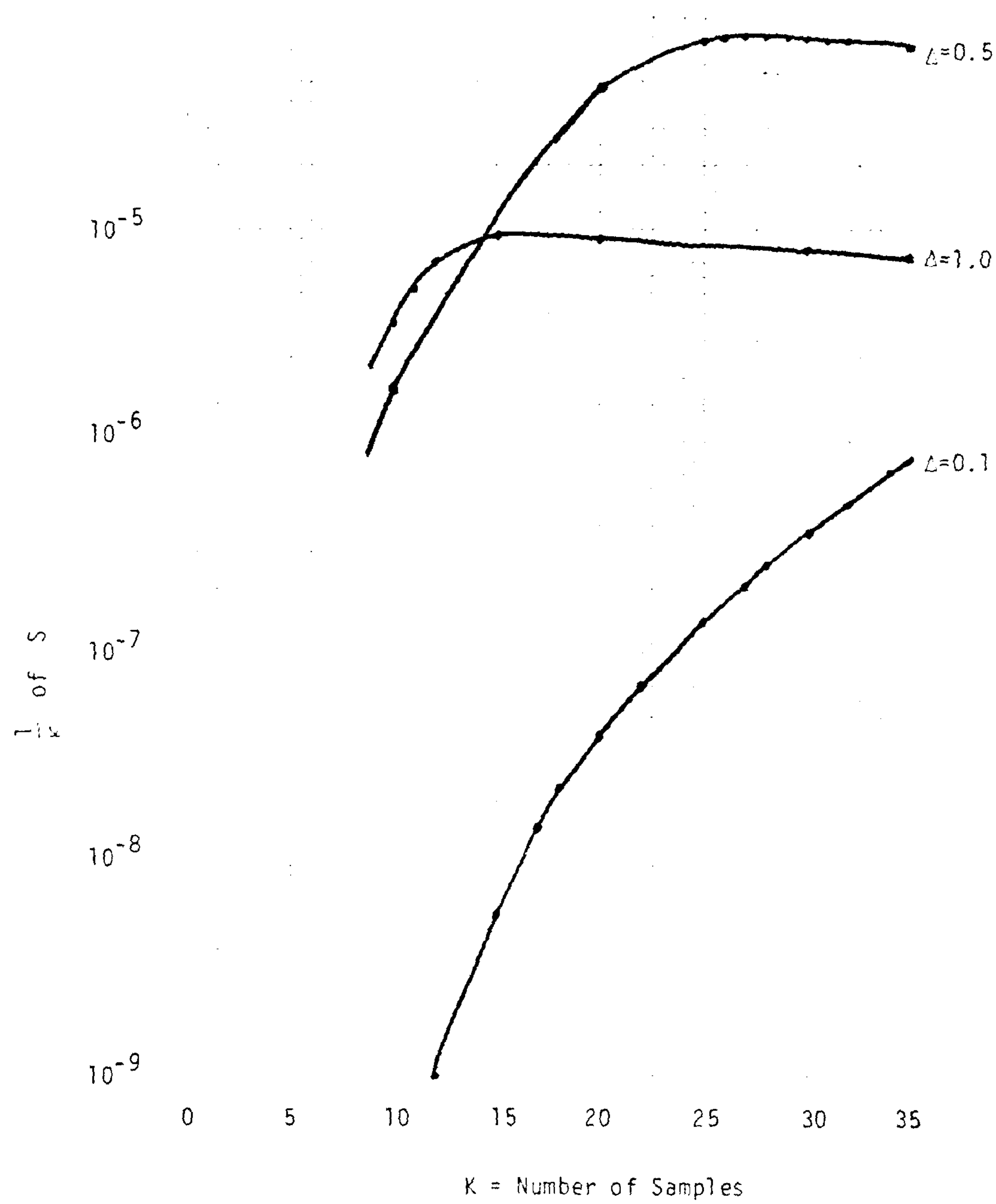

Figure 8. Inverse Condition Number vs. Number of Samples Taken for System of Eqs (86) and (87) with Unit Step Input and Δ = 0.1 , Δ = 0.5 and Δ = 1.0

design engineer may learn that the optimum number of samples is quite large. For $\Delta = 0.1$, for example, the optimum number of samples (greater than 35) overflows the field length of the interactive terminal. While the program dimensions may be increased and the program run on a batch job, this defeats the purpose of having an interactive program.

Summary

In the first section of this chapter, the modal method was verified by comparing its results with those of the "standard" sensitivity system method. The modal method has been used to compare the diagonal and phase variable canonical forms and to get an idea of the effect on the output sensitivity of adding zeros to the system transfer function. In addition, the condition number of the sensitivity matrix has been shown to improve by increasing the number of samples taken up to the point where taking more samples has little effect on the condition number. And it has been shown that by varying the sample spacing in trial runs of the program, an optimal sample spacing may be found for a specified number of samples.

V. Conclusions and Recommendations for Further Research

The modal method derived in Chapter II has been verified by comparing its results with those of the standard "sensitivity system" method. In Chapter III, the computational load of the software implementing the modal algorithm was analyzed. Structural and input design issues have been investigated in Chapter IV.

The following conclusions may now be made:

1. The modal method is computationally more efficient than the standard "sensitivity system" method. The formation of G requires on the order of NA * NPA multiplications, while the E and F matrices require on the order of NA * K multiplications.

2. The modal method is inherently more accurate than the sensitivity system method since it eliminates the numerical problems of integrating "stiff" differential equations.

3. The use of the singular value decomposition in conjunction with the new algorithm leads to a structural condition on parameter identifiability. This is a significant contribution of the algorithm since the structural condition aids in the investigation of canonical forms and in the overall evaluation of system models.

4. For all cases in Chapter IV, the sensitivity matrix of the diagonal form is better conditioned than is that of the phase variable canonical form. This implies that the diagonal canonical form is more accurate in estimating system parameters than is the phase variable form.

5. The interactive program implementing the modal method is very helpful to the designer in both the experimental design phase and the actual parameter estimation task. The program may easily be extended into the iterative algorithm described in Chapter II to perform the entire parameter estimation task.

6. For a constant number of sample times, an optimum sample spacing may be found.

7. For a constant sample spacing, taking more samples increases the accuracy of the parameter estimation, until a point is reached after which taking more samples has little effect on the accuracy.

Although the following areas have not been researched fully, the available software allows their investigation:

1. Input design optimization.
   Optimum sample spacing and number of samples, both separately and in conjunction may be determined. In addition, discrete inputs other than the unit step may be used (Ref 9).

2. Initial state optimization.
   States other than the zero initial state may lead to better accuracy of the parameter estimation.

3. Nth order reduction.
   The relationship between the estimation accuracy of the identifiable parameters and state model order reduction techniques may be investigated (Ref 8).

4. Canonical forms and parameters.
   Canonical forms and system parameters other than the diagonal and phase variable forms investigated in this paper may be chosen to model the system. The special block diagonal form discusse in Chapters III and IV is one example (Ref 17).

5. Numerical problems of close eigenvalues.
   The growing output sensitivities occurring when the eigenvalues of the plant matrix A approach each other (See Eqs (58) and (59).) should be given close attention. This matter is discussed more fully by Reid and Palmer (Ref 17: 21-33).

Bibliography

1. Astrom, K. J. and P. Eykhoff. "System Identification--A Survey," Automatica, 7: 123-162 (1971).

2. Crossley, T. R. and B. Porter. "Eigenvalue and Eigenvector Sensitivities in Linear Systems Theory," Int. J. Control, 10 (2): 163-170 (February 1969).

3. Dongarra, J. J. et al. LINPACK Users' Guide. Philadelphia: Society for Industrial and Applied Mathematics, 1979.

4. Gupta, N. K. and R. K. Mehra. "Computational Aspects of Maximum Likelihood Estimation and Reduction in Sensitivity Function Calculations," IEEE Transactions on Automatic Control, 19 (6): 774-783 (December 1974).

5. IMSL Reference Manual. Houston, Texas: IMSL, Inc., 1979.

6. Kumar, K. S. and R. Sridhar. "On the Identification of Control Systems by the Quasi-Linearization Method," IEEE Transactions on Automatic Control, 22: 242-246 (April 1977).

7. Lawson, C. L. and R. J. Hanson. Solving Least Square Problems. Englewood Cliffs, New Jersey: Prentice-Hall, 1974.

8. McClendon, J. R. Model Order Reduction Using the Balanced State Representation: Theory, Applications and Interactive Software Implementation. MS Thesis. Wright-Patterson AFB, Ohio: Air Force Institute of Technology, December 1979.

9. Mehra, R. K. "Optimal Input Signals for Parameter Estimation in Dynamic Systems--Survey and New Results," IEEE Transactions on Automatic Control, 19 (6): 753-768 (December 1974).

10. Nikolai, Paul J. and Donald S. Clemm. Solution of Ordinary Differential Equations on the CDC 6600/CYBER 74 Processors. Wright-Patterson AFB, Ohio: Air Force Flight Dynamics Laboratory, 1977.

11. Reid, J. G., et al. "Algebraic Representation of Parameter Sensitivities in Linear, Time-Invariant Systems," Journal of the Franklin Institute, 301 (1 and 2): 123-141 (January and February 1976).

12. Reid, J. G. "Structural Identifiability in Linear Time-Invariant Systems," IEEE Transactions on Automatic Control, 22: 242-246 (April 1977).

13. Reid, J. G. Lecture materials destributed in EE 5.10, Linear Systems Analysis and Digital Computation Methods. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 1979.

14. Reid, J. G. A New Parallel Identification Algorithm for Linear Time-Invariant Systems: Preliminary Results. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 1977.

15. Reid, J. G. "Identifiability and Experimental Design Issues in Linear System Identification," ASME Journal: 1 - 10 (December 1979).

16. Reid, J. G. School of Engineering, Air Force Institute of Technology, (personal correspondence). Wright-Patterson AFB, Ohio, May, 1980.

17. Reid, J. G. and L. K. Palmer. A New, Highly Efficient, Parallel Computation Algorithm for the Output Parameter Sensitivities in Linear Time-Invariant Systems. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 1980.

18. Stewart, G. W. "On the Perturbation of Pseudo-Inverses, Projections, and Linear Least Squares Problems," SIAM Review, 19 (4): 634-662 (October 1977).

19. Strang, Gilbert. Linear Algebra and its Applications. New York, San Francisco, and London: Academic Press, 1976.

Appendix A

Derivation of Recursive Formulas for Matrices E and F

The equations for calculating the time- and input-dependent matrices are derived in the following manner. Equation (34) is repeated here:

$$\underline{e}_k' = \left[e^{\lambda_1 t_k} , \ e^{\lambda_2 t_k} , \ldots , e^{\lambda_{NA} t_k} , \ t_k \, e^{\lambda_1 t_k} , \right.$$

$$\left. t_k \, e^{\lambda_2 t_k} , \ldots , t_k \, e^{\lambda_{NA} t_k} \right]_{1 \times 2NA} \qquad (34)$$

When the sample times are all spaced Δ apart, $t_1 = \Delta$, $t_2 = 2\Delta$, and so on until $t_{NA} = NA * \Delta$. Then the first NA columns of E may be computed by the recursive formula

$$E1(k,\ell) = e^{\lambda_\ell \Delta} * E1(k-1,\ell) \qquad (38)$$

with

$$E1(1,\ell) = e^{\lambda_\ell \Delta} \qquad (40)$$

Similarly, the second NA columns of E may be formed by setting

$$E3(k,\ell) = k * \Delta * E1(k,\ell) \qquad (39)$$

with

$$E3(1,\ell) = \Delta * E1(1,\ell) \qquad (41)$$

In matrix F the kth row is given by

$$\underline{f}_k = \Big[e^{\lambda_1 t_k} \int_0^{t_k} e^{-\lambda_1 \tau} U(\tau)\, d\tau \,, \quad e^{\lambda_2 t_k} \int_0^{t_k} e^{-\lambda_2 \tau} U(\tau)\, d\tau \,,$$

$$\ldots, \quad e^{\lambda_{NA} t_k} \int_0^{t_k} e^{-\lambda_{NA} \tau} U(\tau)\, d\tau \,,$$

$$e^{\lambda_1 t_k} \int_0^{t_k} (t_k - \tau)\, e^{-\lambda_1 \tau} U(\tau)\, d\tau \,, \quad e^{\lambda_2 t_k}$$

$$* \int_0^{t_k} (t_k - \tau)\, e^{-\lambda_2 \tau} U(\tau)\, d\tau \,, \ldots,$$

$$e^{\lambda_{NA} t_k} \int_0^{t_k} (t_k - \tau)\, e^{-\lambda_{NA} \tau} U(\tau)\, d\tau \Big] \qquad (38)$$

The first NA columns are given by

$$F1(k,\ell) = e^{\lambda_\ell k\Delta} \int_0^{k\Delta} e^{-\lambda_\ell \tau} U(\tau)\, d\tau \qquad (39)$$

For k = 1,

$$F1(k,1) = e^{\lambda_\ell \Delta} \int_0^{\Delta} e^{-\lambda_\ell \tau} U(\tau)\, d\tau = e^{\lambda_\ell \Delta} \left. \frac{e^{-\lambda_\ell \tau}}{-\lambda_\ell} \right|_0^{\Delta} * U_d(1)$$

$$= C4 * U_d(1) \qquad (40)$$

where $C4 = \dfrac{e^{\lambda_\ell \Delta} - 1}{\lambda_\ell}$. For k = 2, 3,. . ., NA

$$F1(k,\ell) = e^{\lambda_\ell k\Delta} \int_0^{k\Delta} e^{-\lambda_\ell \tau} U(\tau)\, d\tau$$

$$= e^{\lambda_\ell \Delta} \left(e^{\lambda_\ell (k-1)\Delta} \int_0^{(k-1)\Delta} e^{-\lambda_\ell \tau} U(\tau)\, d\tau \right)$$

$$+ e^{\lambda_\ell k\Delta} \int_{(k-1)\Delta}^{k\Delta} e^{-\lambda_\ell \tau} U(\tau)\, d\tau$$

$$= e^{\lambda_\ell \Delta} * F1(k-1,\ell) + e^{\lambda_\ell k\Delta} \left[\frac{e^{-\lambda_\ell \tau}}{-\lambda_\ell} \right]_{(k-1)\Delta}^{k\Delta} * U_d(k)$$

$$= e^{\lambda_\ell \Delta} * F1(k-1,\ell) + C4 * U_d(k) \tag{43}$$

As for the second NA columns of Eq (35), when k=1

$$F3(1,\ell) = e^{\lambda_\ell k\Delta} \int_0^{\Delta} (\Delta - \tau)\, e^{-\lambda_\ell \tau}\, U(\tau)\, d\tau$$

$$= \Delta * \frac{e^{\lambda_\ell \Delta} - 1}{\lambda_\ell} * U_d(1) - e^{\lambda_\ell \Delta} * \left[\int_0^{\Delta} e^{-\lambda_\ell \tau} d\tau \right] * U_d(1)$$

$$= \Delta * \frac{e^{\lambda_\ell \Delta} - 1}{\lambda_\ell} * U_d(1) + e^{\lambda_\ell \Delta} * \left[\frac{(\lambda_\ell \tau + 1) e^{-\lambda_\ell \tau}}{\lambda_\ell^{\,2}} \right]_0^{\Delta} * U_d(1)$$

$$= \left[\frac{\Delta e^{\lambda_\ell \Delta} - \Delta}{\lambda_\ell} - \frac{e^{\lambda_\ell \Delta}}{\lambda_\ell^2} + \frac{\Delta}{\lambda_\ell} + \frac{1}{\lambda_\ell^2} \right] * U_d(1)$$

$$= C5 * U_d(1) \tag{45}$$

where $C5 = \frac{1}{\lambda_\ell^2} + \frac{\Delta e^{\lambda_\ell \Delta}}{\lambda_\ell} - \frac{e^{\lambda \Delta}}{\lambda_\ell^2}$.

For $k = 2, 3, \ldots, NA$, the second NA columns of Eq (35) are given by

$$F3(k,\ell) = e^{\lambda_\ell k\Delta} \int_0^{k\Delta} (k\Delta - \tau)\, e^{-\lambda_\ell \tau}\, U(\tau)\, d\tau$$

$$= e^{\lambda_\ell \Delta} \left(e^{\lambda_\ell (k-1)\Delta} \int_0^{(k-1)\Delta} e^{-\lambda_\ell \tau}\, U(\tau)\, d\tau \right)$$

$$+ \quad e^{\lambda_\ell k\Delta} \int_{(k-1)\Delta}^{k\Delta} (k\Delta - \tau)\, e^{-\lambda_\ell \tau}\, U(\tau)\, d\tau \tag{89}$$

With a little algebraic manipulation, the first term of Eq (89) may be written

$$e^{\lambda_\ell \Delta} \left(e^{\lambda_\ell (k-1)\Delta} \int_0^{(k-1)\Delta} (k\Delta - (k-1)\Delta + (k-1)\Delta - \tau) \right)$$

$$* \quad e^{-\lambda_\ell \tau}\, U(\tau)\, d\tau$$

$$= e^{\lambda_\ell \Delta} \left(e^{\lambda_\ell (k-1)\Delta} \int_0^{(k-1)\Delta} ((k-1)\Delta - \tau)\, e^{-\lambda_\ell \tau} U(\tau)\, d\tau \right.$$

$$\left. + \Delta\, e^{\lambda_\ell (k-1)\Delta} \int_0^{(k-1)\Delta} e^{-\lambda_\ell \tau}\, U(\tau)\, d\tau \right)$$

$$= e^{\lambda_\ell \Delta}\left(F3(k-1,\ell) + \Delta * F1(k-1,\ell)\right) \tag{90}$$

By making the change of variable $\zeta = \tau - (k-1)\Delta$, the second term of Eq (89) becomes

$$e^{\lambda_\ell k\Delta}\int_0^\Delta (k\Delta - \zeta - (k-1)\Delta)\, e^{-\lambda_\ell (k-1)\Delta} * e^{-\lambda_\ell \zeta} U(\zeta-(k-1)\Delta)\, d\zeta$$

$$= e^{\lambda_\ell \Delta}\int_0^\Delta (\Delta - \zeta)\, e^{-\lambda_\ell \zeta}\, d\zeta * U_d(k)$$

$$= C5 * U_d(k) \tag{91}$$

where C5 is defined after Eq (45). Combining the two terms of Eq (89) gives

$$F3(k,\ell) = e^{\lambda_\ell \Delta} * \left(F3(k-1,\ell) + \Delta * F1(k-1,\ell)\right) + C5 * U_d(k) \tag{43}$$

(Ref 17).

The resulting subroutine EFMAT may be seen at the end of the computer program listing in Appendix B.

Appendix B

Computer Program Implementing Modal Method Calculation of the Sensitivity Matrix

The following is the listing of the software used to calculate the sensitivity matrix via the modal method. The software is used on the CDC 6600/CYBER 74. Subroutines EIGRF (or EIGCC), LSVDF and LEQT1C from the IMSL software package are utilized in the calculations (Ref 5). Appendix C contains the user's guide to this program.

```
      PROGRAM LKP(INPUT,OUTPUT)
      COMMON A(10,10)
      COMMON I2(3,10),P(10),IX(2,10),T(10)
      COMMON IE(3,10),D(10),IO(2,10)
      COMMON RK(100),WK1(100)
      COMMON U(30)
      COMMON EI[illegible](10),EI[illegible]V(10,10)
      COMMON EIGI(10,10),REIGV(10,10),CEIGV(10)
      COMMON ZTR(10,10)
      COMMON REIGVX(10),REIGVS(10),CIEIGV(10,10)
      COMMON CEIGVI(10,10),REVIXI(10,10),REVXII(10,10)
      COMMON REVEI(1 ,10),RE IB(10,10)
      COMMON SMAT(20,10),[illegible] IC(40,10)
      COMMON REAL(3 ,10), TOT(40,10),IS(2 )
      COMMON S(10),UT(40,40)
      COMMON UTRANS(3 ,3 ),UTS(2 ,2 )
      COMMON SMATR(30,10),SMATSF(30,10)
      COMMON E(30,20),F(30,20),SMAT(10,10)
      COMMON SMATI(30,10)
      COMMON CN3ZI,CN-ZS,CNS,CNS
      COMPLEX EI[illegible],EI[illegible]V,EIGI,REIGV
      COMPLEX CEIGV,ZTR,REIGVX,REIGV
      COMPLEX CIEIGV,CEIGVI,REVIXI
      COMPLEX REVXII,REVEI,REVIB
      COMPLEX SMAT,[illegible],E,F,SMAT
      COMPLEX SMATI
C
      PRINT*,"TITLE:  [illegible] ALGORITHM FOR CALCULATING THE"
      PRINT*,"        SENSITIVITY MATRIX OF A LINEAR,"
      PRINT*,"        SINGLE-INPUT SINGLE-OUTPUT (SISO)"
      PRINT*,"        TIME-INVARIANT SYSTEM"
      PRINT*," "
      PRINT*,"        [illegible](TIME))/[illegible]=A(P)*X(TIME,P) + B(P)*U(TIME)"
      PRINT*,"        Y(DISCRETE TIME)=    [illegible]* X((DISCRETE TIME)"
      PRINT*," "
      PRINT*,"        [illegible]"
      PRINT*,"        THE [illegible] OF PARAMETERS OF INTEREST"
      PRINT*,"        A,X,[illegible], AND C ARE REAL."
      PRINT*,"IF B,X,[illegible], AND C ARE COMPLEX, TYPE"
      PRINT*,"CR (RETURN) AND CALL PROGRAM LKP."
      PRINT*," "
      PRINT*,"DO YOU WANT TO KNOW"
      PRINT*,"HOW TO USE THIS PROGRAM?"
      PRINT*,"TYPE 1 FOR YES,2 FOR NO."
      PRINT*," "
      READ*,IR1
      IF (IR1.EQ.2) GO TO 3
      PRINT*,"      DEFINITIONS AND COMMENTS"
      PRINT*,"WE ARE ASSUMING THAT WE HAVE GOOD ESTIMATES"
      PRINT*,"OF THE VALUES OF THE ELEMENTS OF A,X,[illegible] C,AND"
      PRINT*,"THAT WE KNOW THE PARAMETER LOCATIONS IN THESE"
      PRINT*,"MATRICES/ [illegible]."
      PRINT*," "
      PRINT*,"FOR SISO SYSTEMS X,[illegible] ARE 1-DIM. ARRAYS."
      PRINT*,"[illegible]"
      PRINT*,"[illegible]"
      PRINT*,"[illegible](5)"
```

```
      PRINT*," "
      PRINT*,"THIS PROGRAM COMPUTES EVAL,EVECT, AND RECIP"
      PRINT*,"EVECT SENSITIVITIES WRT PARAMETERS IN A VIA"
      PRINT*,"METHODS PROPOSED BY R. PORTER AND R. CROSSLEY."
      PRINT*," "
      PRINT*,"SENSITIVITIES OF THE SYSTEM WRT PARAMETERS IN"
      PRINT*,"A,XO,B, AND C ARE ASSUMED TO BE LINEAR."
      PRINT*," "
      PRINT*,"NP=NUMBER OF PARAMETERS        = 10 MAX"
      PRINT*," "
      PRINT*,"NA=DIMENSION OF A"
      PRINT*,"A(NA X NA), SYSTEM PLANT MATRIX"
      PRINT*,"NPA=NO. OF PARAMETERS OF INTEREST IN A"
      PRINT*,"IA(2 X NPA)=LOCATIONS OF PARAMETERS IN A"
      PRINT*," "
      PRINT*,"XO(NA X 1), NOMINAL STATE VECTOR"
      PRINT*,"NPX=NO. OF PARAMETERS OF INTEREST IN XO"
      PRINT*,"IX(NPX)=LOCATIONS OF PARAMETERS IN XO"
      PRINT*," "
      PRINT*,"NB=NO. OF COLUMNS IN B=NO. OF CONTROLS"
      PRINT*,"B(NA X 1),(NA X NB) IN MULTI-INPUT CASE"
      PRINT*,"NPB= NO. OF PARAMETERS OF INTEREST IN B"
      PRINT*,"IB(2 X NPB)=LOCATIONS OF PARAMETERS IN B"
      PRINT*," "
      PRINT*,"NC=NO. OF ROWS OF C=NO. OF OUTPUTS"
      PRINT*,"C(1 X NA),(NC X NA) IN MULTI-OUTPUT CASE"
      PRINT*,"NPC=NO. OF PARAMETERS OF INTEREST IN C"
      PRINT*,"IC(2 X NC)=LOCATIONS OF PARAMETERS IN C"
      PRINT*," "
      PRINT*,"INPUT ALL MATRICES BY COLUMNS!!!"
      PRINT*," "
      PRINT*,"(DO NOT ENTER ANY VALUE(S) FOR:"
      PRINT*,"     IA IF NPA=0"
      PRINT*,"     IX IF NPX=0"
      PRINT*,"     IB IF NPB=0"
      PRINT*,"     IC IF NPC=0)"
      PRINT*,"A,XO,B,C ARE REAL."
      PRINT*,"FIRST COL OF IA SHOULD READ"
      PRINT*,"  IA(1,1)=LOWEST PARAM NO. WHICH APPEARS IN A"
      PRINT*,"  IA(2,1)=ROW ADDRESS OF THIS PARAM. IN A"
      PRINT*,"  IA(3,1)=COL. ADDRESS OF THIS PARAM. IN A"
      PRINT*," "
      PRINT*,"SIMILARLY, IX(1,1)/IB(1,1)/IC(1,1)=LOWEST"
      PRINT*,"  PARAM. NO. WHICH APPEARS IN XO/B/C; AND"
      PRINT*,"  IX(2,1)/IB(2,1)/IC(2,1)=ADDRESS OF THIS"
      PRINT*,"  PARAMETER IN XO/B/C."
      PRINT*," "
      PRINT*,"CEI(J)=C * JTH EVECT"
      PRINT*,"REIX(J)=JTH RECIP EVECT (R.E.) * X VECTOR"
      PRINT*,"REIB(J)=JTH R.E. * B VECTOR"
      PRINT*," "
      PRINT*,"CDIFTL=C * JTH EVECT"
      PRINT*,"CEIV(I)=C * SENSITIVITY OF JTH EVECT WRT"
      PRINT*,"          PARAMETER I"
      PRINT*,"REIXV(I)=JTH R.E. * SENSITIVITY OF X VECTOR WRT"
      PRINT*,"          PARAMETER I"
      PRINT*,"REIVB(I)=SENSITIVITY OF JTH R.E. WRT PARAM I * "
```

```
      PRINT*,"            VECTOR"
      PRINT*,"REVRI=UTH R.S. * SENSITIVITY OF R VECTOR WRT "
      PRINT*,"           PARAMETER I"
      PRINT*,"EIRI= SENSITIVITY OF EIGENVALUES WRT PARAMETER I"
      PRINT*,"COLUMNS OF EACH OF THE ABOVE"
      PRINT*,"MATRICES ARE VECTORS. FOR"
      PRINT*,"EXAMPLE, THE FIRST COLUMN OF"
      PRINT*,"EI,I IS THE SENSITIVITY OF"
      PRINT*,"ALL THE EIGENVALUES WITH RESPECT"
      PRINT*,"TO THE FIRST PARAMETER."
      PRINT*," "
      PRINT*,"D=DELTA=SAMPLE SPACING"
      PRINT*,"K=NUMBER OF DISCRETE TIMES = 3? MAX"
      PRINT*,"U=VECTOR CONTAINING DISCRETE INPUTS"
      PRINT*,"SPRIG=(SZI,SZS), STRUCTURE-DEPENDENT PART OF SMAT"
      PRINT*,"S=TIME-DEPENDENT PART OF SMAT"
      PRINT*,"F=INPUT- AND TIME-DEPENDENT PART OF SMAT"
      PRINT*,"SMAT=(S,F) * SPIG, TOTAL SENSITIVITY MATRIX"
      PRINT*,"  ITH COLUMN=SENSITIVITIES OF (Y MEASURED -"
      PRINT*,"                    Y ESTIMATED) WRT PARAMETER I."
      PRINT*,"  KTH ROW=SENSITIVITIES OF (Y MEASURED -"
      PRINT*,"                  Y ESTIMATED) AT TIME T(K)"
      PRINT*," "
      PRINT*,"CN,ZI,CN,ZS,CNC,CNS= INVERSE CONDITION"
      PRINT*,"NUMBERS OF FZI,FZS,FRIG,SMAT RESPECTIVELY"
      PRINT*," "
      PRINT*,"                              S(NP)/S(1)"
      PRINT*,"WHERE S IS THE VECTOR OF THE NP"
      PRINT*,"SINGULAR VALUES OF FZI,FZS,FRIG, OR SMAT."
      PRINT*,"SMAT=FRIG FOR ZERO STATE CASE."
      PRINT*,"ALL DIMENSIONS ARE THE SAME."
      PRINT*," "
      PRINT*,"UT,UTR, AND UTS= MATRICES OF LEFT"
      PRINT*,"SINGULAR VECTORS OF UTT, FZS & FZI, AND SMAT."
      PRINT*,"VI AND V= MATRICES OF RIGHT SINGULAR VECTORS"
      PRINT*,"OF FZS, ZI, AND RIG & SMAT."
      PRINT*," "
      PRINT*,"ENTER NA,NPA,NPX,NPI,NPS,NP"
      PRINT*," "
      READ*,NA,NPA,NPX,NPI,NPS,NP
      NAT=2 * NA
      NA4=4 * NA
      PRINT*,"IS THE NOMINAL STATE VECTOR"
      PRINT*,"X, EQUAL TO  ?"
      PRINT*,"   1=YES, 2=NO."
      PRINT*," "
      READ*,IRZ
      IF (IRZ.EQ.2) GO TO 4
      CALL ZERO(NAT,NA,1)
      GO TO 5
    4 PRINT*,"ENTER X "
      PRINT*," "
      READ*,(XI(J),J=1,NA)
    5 PRINT*,"ENTER [illegible]"
      PRINT*," "
      DO 1 J=1,NA
      DO 1 I=1,N[illegible]
```

```
 13       READ*,A(I,J)
      IF (NPA.NE.0) GO TO 14
      CALL ZERO(IA,3)
      GO TO 20
 14   DO 15 M=1,NPA
         READ*,IA(1,M),IA(2,M),IA(3,M)
 15   CONTINUE
 20    IF (NPX.NE.0) GO TO 29
      CALL ZERO(IX,2)
      GO TO 35
 29   DO 30 LPX=1,NPX
 30      READ*,IX(1,LPX),IX(2,LPX)
 35      DO 40 LM=1,NA
 40      READ*, (LM)
      IF (NPI.NE.0) GO TO 44
      CALL ZERO(II,2)
      GO TO 50
 44   DO 45 LPI=1,NPI
 45   READ*,II(1,LPI),II(2,LPI)
 50   DO 55 LA=1,NA
 55      READ*,D(LA)
      IF (NPD.NE.0) GO TO 59
      CALL ZERO(ID,2)
      GO TO 65
 59   DO 60 LPD=1,NPD
 60      READ*,ID(1,LPD),ID(2,LPD)
C
 65   DO 70 J=1,NA
         EIG(J)=0.
         DO 70 I=1,NA
         EIGVT(I,J)=0.
         EIGVX(I,J)=0.
         IF(I.EQ.J) EIGVX(I,J)=1.
 70      CONTINUE
C
C        EIGVT IS A EIGENVECTOR MATRIX: VALUES ARE TRANSPOSE
C         OF THOSE IN EIGVT MATRIX IN PROGRAM LKP
C        (SEE COMMENTS IN STEP 1, BELOW)
      ITR=[illegible]
C
C          STEP 1, PART A
C          --------------
C
C     [illegible] EIGENVALUES OF A
C     EIGENVECTORS OF [illegible]
C     [illegible] MATRIX [illegible] IS A; EACH COLUMN IS AN VECT
      CALL EIGRF(A,NA,IA,0,EIG,EIGVT,IA,WK,IER)
C     [illegible] IER
      DO 75 J=1,NA
         DO 75 I=1,NA
         [illegible](J,I)=EIGVT(I,J)
 75   CONTINUE
C
C          STEP 1, PART B
C          --------------
C
C     [illegible] ALL [illegible]
```

```
      C        CHECK FOR REPEAT EVAL
      C          ALSO CHECK FOR COMPLEX EVAL
      C                           (0 IF EIG(I) IS REAL
      C
      C        IS(I)=IS(NA+I)= (1 IF EIG(I) IS
      C                           FIRST OF COMPLEX
      C                           CONJUGATE PAIR
      C
      C                           (2 IF EIG(I) IS SECOND OF
      C                            COMPLEX CONJUGATE PAIR
      C
               CALL ZERO1(IS,II)
               II=NA + 1
               DO 80 I=1,NA
                  IF(EIG(I).NE.0.)GO TO 78
                  PRINT*,"EIGENVALUES CAN'T EQUAL 0."
                  PRINT*,"YOUR EIGENVALUE NUMBER ",I
                  PRINT*," EQUALS 0."
                  STOP "TRY DIFFERENT A MATRIX."
         78       IF(AIMAG(EIG(I)).EQ.0.) GO TO 79
                  IS(I)=1
                  IF (I.EQ.1) GO TO 79
                  IF (IS(I-1).EQ.1) IS(I)=2
         79       IS(II)=IS(I)
                  II=II + 1
                  DO 80 J=1,[illegible]
                  IF (I.EQ.J) GO TO 80
                  IF (EIG(I).[illegible].EIG(J)) GO TO [illegible]
               PRINT*,"[illegible] REPEATED EIGENVALUES."
               PRINT*,"[illegible]"
               PRINT*," [illegible] DISTINCT EIGENVALUES."
               [illegible]
      C
      C
      C
      C
      C           [illegible]
      C           --------------
      C
      C        FIND RECIPROCAL [illegible]
      C        [illegible] EA COL IS A VECTOR
               CALL [illegible](ZZR,[illegible].REIGV.NA.II,IDGT,WK,IER)
      C
               PRINT*,"ZE"
               DO 85 I=1,NA
                  PRINT*,(Z(I,J),J=1,NA)
         85    CONTINUE
               PRINT*," "
               PRINT*,"X = "
               PRINT*,(X(J),J=1,NX)
               PRINT*," "
               PRINT*,"Y = "
               PRINT*,(Y(J),J=1,[illegible])
               PRINT*," "
               PRINT*,"[illegible] "
               PRINT*,([illegible](J),J=1,[illegible])
               PRINT*," "
               PRINT*,"[illegible] "
```

```
      DO 100 J=1,3
  100    PRINT*,(IA(J,I),I=1,NPA)
      PRINT*," "
      PRINT*,"IX= "
      DO 105 J=1,3
  105    PRINT*,(IX(J,I),I=1,NPX)
      PRINT*," "
      PRINT*,"IL= "
      DO 110 J=1,3
  110    PRINT*,(IL(J,I),I=1,NPL)
      PRINT*," "
      PRINT*,"IC= "
      DO 115 J=1,3
  115    PRINT*,(IC(J,I),I=1,NPC)
      PRINT*," "
      PRINT*,"NP= ",NP
      PRINT*," "
      PRINT*,"EIG= "
      DO 120 I=1,NA
  120    PRINT*,EIG(I)
      PRINT*," "
      PRINT*,"EIGV= "
      DO 125 I=1,NA
         PRINT*,(EIGV(I,J),J=1,NA)
  125 CONTINUE
      PRINT*," "
      PRINT*,"REIGV= "
      DO 130 I=1,NA
         PRINT*,(REIGV(I,J),J=1,NA)
  130 CONTINUE
C        STEP 3
C        ------
C
C        [illegible]
C
      [illegible] NOT [illegible] PARAMETERS:
      [illegible]
      [illegible]
      IF (IPR.EQ.0) GO TO 131
C     IF [illegible]
C
      CALL ZERO([illegible],NA,1)
C
C     IF [illegible] MULTIPLY [illegible]
  131 DO 133 J=1,NA
         [illegible](J)=0.
         DO 132 LA=1,NA
  132    [illegible](J)=[illegible](LA,J) * [illegible](LA) + [illegible](J)
      DO 133 J=1,NA
      [illegible](J)=0.
      [illegible](J)=0.
         DO 133 L=1,NV
C
C     [illegible]
C
      [illegible](J)= [illegible] (J)
C
```

```
      C     [illegible]
      C
                  RETCVT(J)=RETC(LA,J) * T(LA) + RETCVR(J)
      137   CONTINUE
            PRINT*,"WOULD YOU LIKE TO SEE THE CONSTANT"
            PRINT*,"VECTORS?  1=YES, 2=NO."
            PRINT*," "
            READ*,IRE
            IF (IRE.EQ.2) GO TO 134
            PRINT*,"RETCVX= ",(RETCVX(J),J=1,NA)
            PRINT*,"RETCV= ",(RETCV(J),J=1,NA)
            PRINT*,"RETCVB= ",(RETCVB(J),J=1,NA)
      C
      C                 SENSITIVITIES
      C                 -------------
      C
      134   IF(NP)135,140,145
      135    PRINT*,"NUMBER OF PARAMETERS CANNOT BE
           1, LESS THAN [illegible] [illegible] AGAIN."
            STOP       "NP.LT. "
      140   STOP       "NP.EQ."
      145   LP1=1
            LPY=1
            LP2=1
            LP3=1
            CALL [illegible]
            CALL [illegible]
            CALL [illegible]
            CALL [illegible]
            CALL [illegible]
            CALL [illegible]
            CALL [illegible]
            DO [illegible]
            CALL [illegible]
           1[illegible]
           2[illegible](1,I),DIFF[illegible](1,I)
           3,LT[illegible],LP[illegible],LP[illegible])
      1[illegible] CONTINUE
            PRINT*,"WOULD YOU LIKE TO SEE THE"
            PRINT*,"PARAMETER-SENSITIVE MATRICES?"
            PRINT*,"    1=YES, 2=NO."
            PRINT*," "
            READ*,IRS
            IF (IRS.EQ.2) GO TO 1[illegible]
            PRINT*," "
            PRINT*,"[illegible] = "
            DO 1[illegible] J=1,N[illegible]
      1[illegible]    PRINT*,([illegible](J,I),I=1,NP)
            PRINT*," "
            PRINT*,"[illegible] = "
            DO 1[illegible] J=1,N[illegible]
      1[illegible]    PRINT*,([illegible](J,I),I=1,NP)
            PRINT*," "
            PRINT*,"[illegible] = "
            DO 1[illegible] J=1,N[illegible]
      1[illegible]    PRINT*,([illegible](J,I),I=1,[illegible])
            PRINT*," "
```

```
      PRINT*,"REVII= "
      DO 154 J=1,NA
154      PRINT*,(REVTI(J,I),I=1,NP)
      PRINT*," "
      PRINT*,"RINVII= "
      DO 155 J=1,NA
155      PRINT*,(RINVII(J,I),I=1,NP)
      PRINT*," "
      PRINT*,"REVIIE "
      DO 156 J=1,NA
156      PRINT*,(REVII(J,I),I=1,NP)
      PRINT*," "
      PRINT*,"DIEIGV= "
      DO 157 J=1,NA
157      PRINT*,(DIEIGV(J,I),I=1,NP)
C
C          STEP 7
C          ------
C
C     FORM SZI
C     SZI PART OF SENSITIVITY MATRIX WHICH DEPENDS ON
C     THE MATRIX STRUCTURE OF THE PROBLEM (A,XO,_, & D).
C     ZI STANDS FOR ZERO INPUT.
C        R STANDS FOR REAL.
178   CALL ZERO(ZMAT,[illegible],NP)
      IF (IPR.EQ.1) GO TO 178
      CALL SMAT(NA,NAD,NP,EIGV,DEIGVI,EIGNY
     D,[illegible],ZMAT,CIPI(,REVIK))
      PRINT*,"SZI= "
      DO 158 J=1,NAD
         PRINT*," "
158      PRINT*,(ZMAT(J,I),I=1,NP)
      CALL ZERO(ZMAT,[illegible])
      CALL ZER([illegible])
      DO 160 I=1,NP
         DO 160 J=1,NAD
160      ZMAT(J,I)=ZMAT(J,I)
      CALL REALY(I,NAD,NP,ZMAT,REAL)
      DO 161 I=1,NP
         DO 161 J=1,NAD
161   SZI(J,I)=REAL(J,I)
      PRINT*,"DO YOU WANT A SINGULAR VALUE"
      PRINT*,"VALUE DECOMPOSITION OF SZI?"
      PRINT*,"   1=YES,2=NO."
      READ*,IRS
      IF (IRS.EQ.2) GO TO 169
      DO 162 J=1,NAD
         DO 162 K=1,NAD
         UT(J,K)=0.
         IF (K.EQ.J) UT(J,K)=1.
162   CONTINUE
      CALL ZERO(V,NP)
      CALL LSVRF(NPS,[illegible],NAD,NP,[illegible]
     *,[illegible],NAD,[illegible],[illegible])
      PRINT*," "
      PRINT*,"[illegible]"
      PRINT*,"   [illegible] = [illegible] * [illegible]"
```

```
      PRINT*,"TYPE 1 IF YOU WANT TO SEE"
      PRINT*,"JUST THE SINGULAR VALUES (S)"
      PRINT*,"AND CONDITION NUMBER OF IZI"
      PRINT*,"(CNDZI), TYPE 10 IF YOU ALSO"
      PRINT*,"WANT TO SEE THE SING. VECTORS"
      PRINT*,"ENTER 1 OR 2"
      PRINT*," "
      READ*,IR13
      IF (IR13.EQ.1) GO TO 166
      PRINT*," "
      PRINT*,"UTS= "
      DO 163 J=1,NA2
         PRINT*," "
163      PRINT*,(UTS(J,JJ),JJ=1,NA2)
      PRINT*," "
      PRINT*,"VS= "
      DO 164 II=1,NP
         PRINT*," "
164      PRINT*,(VREAL(II,I),I=1,NP)
166   PRINT*," "
      PRINT*,"S= "
         PRINT*,(S(I),I=1,NP)
      CNDZI=S(NP)/S(1)
      PRINT*,"CNDZI= ",CNDZI
      CNDZI=S(NP)/S(1)
C
C            STEP B 1
C            --------
C
C     FROM [illegible]
C     [illegible] STATE.
C     [illegible] FOR [illegible].
1[illegible] CALL [illegible]
      CALL [illegible]
     [illegible]
      NJ=NA2 + 1
      DO 177 J=1,NA2
         DO 17[illegible] I=1,[illegible]
         [illegible](NJ,I)=[illegible](J,I)
17[illegible]    CONTINUE
177   NJ=NJ + 1
      PRINT*,"[illegible]="
      DO 174 J=1,NA4
      PRINT*," "
174   PRINT*,([illegible](J,I),I=1,NP)
      [illegible]
C
C
C            STEP [illegible]
C            --------
C
C     IF [illegible] ONLY [illegible]
175   CALL [illegible]
     [illegible]
      PRINT*," "
      PRINT*,"[illegible]= "
      DO [illegible] J=1,[illegible]
```

```
            PRINT*," "
      176   PRINT*,(GMAT(J,I),I=1,NP)
C
C              STEP 9 A
C              --------
C
C     GET SINGULAR VALUES OF GZS ONLY
  177 PRINT*,"SINCE Y1=1,GZI=1"
      PRINT*,"IF YOU WANT A SINGULAR VALUE"
      PRINT*,"DECOMPOSITION OF GZS?"
      PRINT*,"    1=YES,2=NO."
      PRINT*," "
      READ*,IR6
      IF (IR6.EQ.2) GO TO 185
      CALL REALLY(IS,NA2,NP,GMAT,GREAL)
      DO 178 J=1,NA2
         DO 178 K=1,NA2
         UTS(J,K)=0.
         IF (K.EQ.J) UTS(J,K)=1.
  178 CONTINUE
      CALL PER(P10,NP)
      CALL LSVDF(GREAL,21,NA2,NP,UTS
     C,21,NA2,S,WK,IER)
      PRINT*," "
      PRINT*,"AFTER TAKING SING.VAL.,GZS="
      PRINT*,"    UTS * S * VS"
      PRINT*,"TYPE 1 IF YOU WANT TO SEE"
      PRINT*,"JUST THE SINGULAR VALUES (S)"
      PRINT*,"AND 1/CONDITION NUMBER OF GZS"
      PRINT*,"(C.GZS).TYPE 2 IF YOU ALSO"
      PRINT*,"WANT TO SEE THE SING.VECTORS"
      PRINT*,"[illegible]"
      PRINT*," "
      READ*,IR17
      IF (IR17.EQ.1) GO TO 18[illegible]
      PRINT*," "
      PRINT*,"UTS= "
      DO 181 J=1,NA2
         PRINT*," "
  181    PRINT*,(UTS(J,JJ),JJ=1,NA2)
      PRINT*," "
      PRINT*,"VS= "
      DO 182 II=1,NP
         PRINT*," "
  182    PRINT*,(GREAL(II,I),I=1,NP)
  183 PRINT*," "
      PRINT*,"S= "
         PRINT*,(S(I),I=1,NP)
      C1GZS=S(NP)/S(1)
      PRINT*,"C1GZS= ",C1GZS
      GO TO 185
C
C              STEP 9
C              --------
C
C     GET SINGULAR VALUES OF [illegible]
  18[illegible] PRINT*,"IF YOU WANT A SINGULAR VALUE"
```

```
      PRINT*,"IF COMPOSITION OF G=ZI,Z7S?"
      PRINT*,"    1=YES,2=NO."
      PRINT*," "
      READ*,I27
      IF (I27.EQ.2) GO TO 135
      CALL REALLY(I3,NA2,NP,CMAT,CREAL)
      DO 137 I=1,NP
         DO 137 J=1,NA2
            NJ=J + NA2
137      GTOT(NJ,I)=-REAL(J,I)
      DO 133 J=1,NA4
         DO 133 K=1,NA4
         UT(J,K)=0.
         IF(J.EQ.K) UT(J,K)=1.
133   CONTINUE
      CALL ZER(R,NP)
      CALL LSVIF(GTOT,...,NA4,NP,UT
     C,4C,NA4,S,WK4,IER)
      PRINT*," "
      PRINT*,"AFTER TAKING SING. AL.. GTOT= "
      PRINT*,"     UT * ... "
      PRINT*,"TYPE 1 IF YOU WANT TO SEE"
      PRINT*,"JUST THE SINGULAR VALUES (S)"
      PRINT*,"AND 1/CONDITION NUMBER OF G (CNS)"
      PRINT*,"TYPE 2 IF YOU ALSO WANT TO SEE THE"
      PRINT*,"SINGULAR VECTORS.  ENTER 1 OR 2"
      PRINT*," "
      READ*,IP14
      IF (IP14.EQ.1) GO TO 132
      PRINT*," "
      PRINT*,"UT= "
      DO 131 J=1,NA4
         PRINT*," "
13       PRINT*,(UT(J,JJ),JJ=1,NA4)
      PRINT*," "
      PRINT*,"V= "
      DO 131 II=1,NP
         PRINT*," "
131      PRINT*,(GTOT(II,I),I=1,NP)
132   PRINT*," "
      PRINT*,"S= "
      PRINT*,(S(I),I=1,NP)
      CN=S(NP)/S(1)
      PRINT*,"CNS= ",CN
C
C        STEP 1
C        ------
C
C     FORMATION OF E AND F MATRICES
C
135   PRINT*,"D=SAMPLE PERIOD.  ENTER D."
      PRINT*," "
      READ*,D
      PRINT*,"K=THE NUMBER OF DISCRETE INPUT/OUTPUT TIMES."
      PRINT*,"ENTER KIN, UP TO ..."
      PRINT*," "
      READ*,KIN
      PRINT*,"...,KIN,... DISCRETE INPUTS."
```

```
      PRINT*,"ENTER INPUTS"
      PRINT*," "
      READ*,(U(K),K=1,KIN)
      DO 210 L=1,NA
      CALL ETMAT(E(1,L),E(1,L+NA),F(1,L),F(1,L+NA)
     C,ETS(L),E,U,KIN,NA)
  210 CONTINUE
      PRINT*," DO YOU WANT TO SEE E AND F?"
      PRINT*,"   1=YES, 2=NO."
      PRINT*," "
      READ*,IR12
      IF (IR12.EQ.2) GO TO 212
      PRINT*,"E= "
      DO 215 K=1,KIN
        PRINT*," "
  215   PRINT*,(E(K,L),L=1,NA2)
      PRINT*," "
      PRINT*,"F= "
      DO 211 K=1,KIN
        PRINT*," "
  211   PRINT*,(F(K,L),L=1,NA2)
  212 PRINT*," "
      IF (IR2.EQ.2) GO TO 204
C
C            STEP 11 B
C            ---------
C
C     FORM SENSITIVITY MATRIX OF
C     ZERO STATE SYSTEM
C
      PRINT*,"SMATS= "
      DO 213 I=1,NP
        DO 213 K=1,KIN
        SMATS(K,I)=0.
          DO 213 J=1,NA2
          SMATS(K,I)=F(K,J) * SMAT(J,I) + SMATS(K,I)
  213 CONTINUE
      DO 216 K=1,KIN
        PRINT*," "
  216   PRINT*,(SMATS(K,I),I=1,NP)
      PRINT*,"DO YOU WANT A SINGULAR VALUE"
      PRINT*,"DECOMPOSITION OF SMATS?"
      PRINT*,"   1=YES, 2=NO."
      PRINT*," "
      READ*,IR8
      PRINT*,"DO YOU WANT TO CHANGE JUST THE INPUT?"
      PRINT*,"   1=YES, 2=NO."
      PRINT*," "
      READ*,IR11
      IF (IR8.EQ.1) GO TO 217
      IF (IR11.EQ.2) STOP "THE END"
      GO TO 105
  217 DO 218 I=1,NP
        DO 218 K=1,KIN
        SMATD(K,I)=DBLE(SMATS(K,I))
  218 CONTINUE
      DO 219 J=1,KIN
```

```
            DO 210 I=1,KIN
              UTRANS(I,J)=0.
      210     IF(I.EQ.J) UTRANS(I,J)=1.
            CALL ZERO(S,NP)
            CALL LSVRF(SMATSR,30,KIN,NP,UTRANS
           C,30,KIN,S,WKA,IER)
            PRINT*,"AFTER TAKING SING. VAL.,"
            PRINT*,"   SMATSR=UTRANS * S * V"
            PRINT*," "
            PRINT*,"TYPE 1 IF YOU WANT TO SEE"
            PRINT*,"JUST THE SINGULAR VALUES (S)"
            PRINT*,"AND 1/CONDITION NUMBER OF SMATS (CNS)"
            PRINT*,"TYPE 2 IF YOU ALSO WANT TO SEE THE"
            PRINT*,"SINGULAR VECTORS.  ENTER 1 OR 2."
            READ*,IRIS
              PRINT*," "
            IF (IRIS.EQ.1) GO TO 222
            PRINT*," "
            PRINT*,"UTRANS= "
            DO 220 K=1,KIN
              PRINT*," "
      220     PRINT*,(UTRANS(K,KK),KK=1,KIN)
            PRINT*," "
            PRINT*,"V= "
            DO 221 II=1,NP
              PRINT*," "
      221     PRINT*,(SMATSR(II,I),I=1,NP)
      222   PRINT*," "
            PRINT*,"S= "
            PRINT*,(S(I),I=1,NP)
            CNS=S(NP)/S(1)
            PRINT*,"CNS= ",CNS
            IF (IRIS.EQ.2)STOP "[illegible] INPUT ([illegible] STATE"
            GO TO 14[illegible]
      C
      C         STEP 11 .
      C         --------
      C     FORM SENSITIVITY MATRIX WHERE
      C     INPUTS & INITIAL STATES ARE
      C     NONZERO.
      224   PRINT*,"SMAT= "
            DO 225 I=1,NP
              DO 225 K=1,KIN
              SMAT(K,I)=0.
                DO 227 J=1,NPI
                SMAT(K,I)=F(K,J) * [illegible] IS(J,I) +
           C                  F(K,J) * [illegible] IS(J+NA,I) +
           C                  SMAT(K,I)
      227   CONTINUE
            DO 226 K=1,KIN
              PRINT*," "
      226     PRINT*,(SMAT(K,I),I=1,NP)
            PRINT*,"DO YOU WANT A SINGULAR VALUE"
            PRINT*,"DECOMPOSITION OF SMAT?"
            PRINT*,"[illegible] (YES=1,NO=2)."
            PRINT*," "
            READ*,ID[illegible]
```

```
      PRINT*,"DO YOU WANT TO CHANGE JUST THE INPUT?"
      PRINT*,"    1=YES,  2=NO."
      PRINT*," "
      READ*,IR1
      IF (IR3.EQ.1) GO TO 225
      IF (IR1 .EQ.1) GO TO 195
      STOP "NO MORE INPUT,YOU SAID."
  228 DO 230 I=1,NP
        DO 230 K=1,KIN
          SMATR(K,I)=REAL(SMAT(K,I))
  230   CONTINUE
      DO 231 J=1,KIN
        DO 231 K=1,KIN
          UTRANS(K,J)=0.
  231   IF (K.EQ.J) UTRANS(K,J)=1.
      CALL ZER(RV,3,NP)
      CALL LSVDF(SMATR,30,KIN,NP,UTRANS
     C,30,KIN,S,WKA,IER)
      PRINT*,"AFTER TAKING SING.VALUES."
      PRINT*,"    SMATR=UTRANS * S * V"
      PRINT*,"TYPE 1 IF YOU WANT TO SEE"
      PRINT*,"JUST THE SINGULAR VALUES (S)"
      PRINT*,"TYPE 2 IF YOU ALSO WANT TO SEE"
      PRINT*,"THE SINGULAR VECTORS. ENTER 1 OR 2."
      PRINT*," "
      READ*,IR1A
      IF (IR1A.EQ.1) GO TO 238
      PRINT*,"UTRANS= "
      DO 233 K=1,KIN
        PRINT*," "
  233   PRINT*,(UTRANS(K,KK),KK=1,KIN)
      PRINT*," "
      PRINT*,"V= "
      DO 237 II=1,NP
        PRINT*," "
  237   PRINT*,(SMATR(II,I),I=1,NP)
  238 PRINT*," "
        PRINT*,(S(I),I=1,NP)
      CNS=S(1)/S(2)
      PRINT*,"CNS= ",CNS
      IF (IR11.EQ.1) GO TO 195
      END
```



```
      SUBROUTINE ZER(ZMAT,NR,NT)
      COMPLEX ZMAT(1 , 1)
      DO 5000 JJ=1,NT
        DO 5000 II=1,NR
 5000     ZMAT(II,JJ)=0.
      RETURN
      END
```

```
      SUBROUTINE SENS(I,NPA,NPX,NPB,NPC,IA,IX,IB,IC
     C,NA,EIG,EIGV,REIGV,X1,B,C,EIGI,REIGVI,REVIX1
     C,REVIB,REVX1I,REVC1,DIEIGV,LPA,LPX,LPB,LPC)
      DIMENSION IA(3,NPA),X1(NA),IX(2,NPX),B(NA)
      DIMENSION IB(2,NPB),C(NA),IC(2,NPC)
      COMPLEX EIG(NA),EIGI(NA),EIGV(10,NA),REIGV(10,NA)
      COMPLEX DIEIGV(NA),DEIGVI(NA),REVX1I(NA)
      COMPLEX REVIB(NA),REVBI(NA)
      COMPLEX REVIX1(NA)
      COMPLEX EIGVI(10),REIGVI(10)
      COMPLEX HKJ,HJK
C
C     STEP 3
C     ------
C
C     SENSITIVITIES W.R.T. PARAMETERS IN A
C
      IF(NPA.EQ.0) GO TO 1030
 1000 IF(LPA.GT.NPA) GO TO 1030
      IF(I.LT.IA(1,LPA)) GO TO 1030
C     FORM EIGENVAL SENSITIVITIES WRT PARAM IN A
C     EA COL IS SENSIT. OF THE VECTOR OF EVAL WRT PARAM IN A
      DO 1025 J=1,NA
          EIGI(J)=REIGV(IA(2,LPA),J) * EIGV(IA(3,LPA),J)
C     FORM DEIGVI NA X NP
C     FORM REVIX NA X NP
C     FORM REVIB NA X NP
      REVIX1(J)=0.
      DEIGVI(J)=0.
      REVIB(J)=0.
          DO 1005 LA=1,NA
          EIGVI(LA)=(0.,0.)
 1005     REIGVI(LA)=(0.,0.)
          DO 1015 K=1,NA
          IF(K.EQ.J) GO TO 1015
          HKJ=REIGV(IA(2,LPA),K) * EIGV(IA(3,LPA),J)/(EIG(J)
     C    -EIG(K))
          HJK=REIGV(IA(2,LPA),J) * EIGV(IA(3,LPA),K)/(EIG(J)
     C    -EIG(K))
            DO 1010 LA=1,NA
            EIGVI(LA)=HKJ * EIGV(LA,K) + EIGVI(LA)
            REIGVI(LA)=HJK * REIGV(LA,K) + REIGVI(LA)
 1010       CONTINUE
 1015     CONTINUE
C
        DO 1020 LA=1,NA
        DEIGVI(J)=C(LA) * EIGVI(LA) + DEIGVI(J)
        REVIX1(J)=REIGVI(LA) * X1(LA) + REVIX1(J)
        REVIB(J)=REIGVI(LA) * B(LA) + REVIB(J)
 1020   CONTINUE
 1025 CONTINUE
      LPA=LPA + 1
      GO TO 1000
C
C     STEP 4
C     ------
C
```

```
SUBROUTINE SENS          74/74    OPT=1                                   FTN 4.8+5?1

      C       SENSITIVITIES W.R.T. PARAMETERS IN X
      C
      1030   IF(NPX.EQ.0) GO TO 1060
      C      FORM REVX(I)NA X NP
      1040   IF(LPX.GT.NPX) GO TO 1060
             IF(I.LT.IX(1,LPX)) GO TO 1060
             DO 1050 J=1,NA
             REVXI(J)=REIGV(IX(2,LPX),J)
      1050   CONTINUE
             LPX=LPX + 1
             GO TO 1040
      C
      C           STEP 3
      C           -----
      C
      C       SENSITIVITIES W.R.T. PARAMETERS IN [illegible]
      C
      1060   IF(NPB.EQ.0) GO TO 1090
      1070   IF(LPB.GT.NPB) GO TO 1090
             IF(I.LT.IB(1,LPB)) GO TO 1090
      C      FORM REVB(I)NA X NP
             DO 1080 J=1,NA
             REVBI(J)=REIGV(IB(2,LPB),J)
      1080   CONTINUE
             LPB=LPB + 1
             GO TO 1070
      C
      C           STEP 4
      C           ------
      C
      C       SENSITIVITIES W.R.T. PARAMETERS IN C
      C
      1090   IF(NPC.EQ.0) GO TO 1110
      1100   IF(LPC.GT.NPC) GO TO 1110
             IF(I.LT.IC(1,LPC)) GO TO 1110
      C
      C      FORM REVC(I)NA X NP
             DO 1105 J=1,NA
             REVCI(J)=REIGV(IC(2,LPC),J)
      1105   CONTINUE
             LPC=LPC + 1
             GO TO 1100
      1110   RETURN
             END

SUBROUTINE ZERO2         74/74    OPT=1                                   FTN 4.8+5

             SUBROUTINE ZERO2(ZMAT,NE,NT)
             DOUBLE ZMAT(NE,NT)
             DO 7000 J=1,NT
                DO 7000 I=1,NE
      7000      ZMAT(I,J)=0.
             RETURN
             END
```

```
      SUBROUTINE GMATR(NA,N[illegible],NP,CEIGV,CEIGVI,REVX[illegible]
     C,REVXBI,EIMI,GMAT,CEIIGV,REVIXB)
      COMPLEX GMAT(2[illegible],NP),CEIGV(NA),CEIGVI(1[illegible],NP)
      COMPLEX REVX[illegible](NA),REVXBI(1[illegible],NP),EIMI(1[illegible],NP)
      COMPLEX CEIIGV(1[illegible],NP)
      COMPLEX REVIXB(1[illegible],NP)
      DO 2010 I=1,NP
        DO 2010 J=1,NA
        GMAT(J,I)=(CEIGVI(J,I) + CEIIGV(J,I)) * REVX[illegible](J)
     C        + CEIGV(J) * (REVIXB(J,I) + REVXBI(J,I))
        GMAT(J+NA,I)=EIMI(J,I) * CEIGV(J) * REVXB(J)
 2010 CONTINUE
      RETURN
      END
```



```
      SUBROUTINE REALLY(IS,NA2,NP,GMAT,GREAL)
      DIMENSION IS(NA2),GREAL(2[illegible],NP)
      COMPLEX GMAT(2[illegible],NP)
      DO 2040 J=1,NA2
      IF (IS(J) - 1) [illegible],[illegible],2040
 [illegible] DO [illegible] I=1,NP
          GREAL(J,I)=REAL(GMAT(J,I))
 [illegible]   CONTINUE
          GO TO [illegible]
 [illegible]   DO [illegible] I=1,NP
          GREAL(J,I)=REAL(GMAT(J,I))
          J1=J + 1
          GREAL(J1,I)=AIMAG(GMAT(J,I))
 [illegible]   CONTINUE
 [illegible] CONTINUE
      RETURN
      END
```



```
      SUBROUTINE ZEROA(ZMAT,N[illegible],NP)
      COMPLEX ZMAT([illegible],NP)
      DO [illegible] JJ=1,NP
        DO [illegible] II=1,N[illegible]
 [illegible] ZMAT(II,JJ)=0.
      RETURN
      END
```

```
SUBROUTINE ZEROR        74/74    OPT=1                                   FTN 4.8+[illegible]

      SUBROUTINE ZEROR(RZMAT,NN)
      DIMENSION RZMAT(NN)
      DO 5000 II=1,NN
 5000    RZMAT(II)=0.
      RETURN
      END
```

```
SUBROUTINE ZEROI        74/74    OPT=1                                   FTN 4.8+[illegible]

      SUBROUTINE ZEROI(IZMAT,NN)
      DIMENSION IZMAT(NN)
      DO 4000 II=1,NN
 4000    IZMAT(II)=0
      RETURN
      END
```

```
SUBROUTINE [illegible]EMAT        74/74    OPT=1                             FTN 4.8+[illegible]

      SUBROUTINE [illegible]EMAT([illegible])
      DIMENSION U(N)
      COMPLEX E1(NA),E3(NA),F1(NA),F3(NA),[illegible]
      COMPLEX EL,C ,C1,C2,C3,C4,C[illegible]
      EL=CEXP([illegible])
      C1=1./EL
      C2=C1*[illegible]
      C3=EL*C
      C3=C3*C[illegible]
      C4=C2-C.
      C5=C1+C*C2-C3
      E1(1)=EL
      E3(1)=C*EL
      F1(1)=C4*U(1)
      F3(1)=C5*U(1)
      DO 7000 K=2,N
         E1(K)=EL*E1(K-1)
         E3(K)=C*K*E1(K)
         F1(K)=C1*F1(K-1)+C4*U(K)
         F3(K)=EL*(F3(K-1)+C*F1(K-1))+C5*U(K)
 7000 CONTINUE
      RETURN
      END
```

Appendix C

User's Guide to Interactive Program Implementing Modal Method Calculation of the Sensitivity Matrix

Program LKP (or LKPC) computes the sensitivity matrix

$$S = \begin{bmatrix} y_{(1)}(t_1) & y_{(2)}(t_1) & \cdots & y_{(NP)}(t_1) \\ y_{(1)}(t_2) & y_{(2)}(t_2) & \cdots & y_{(NP)}(t_2) \\ \vdots & \vdots & & \vdots \\ y_{(1)}(t_K) & y_{(2)}(t_K) & \cdots & y_{(NP)}(t_K) \end{bmatrix}_{K \times NP} \qquad (8)$$

of a linear, single-input single-output, time-invariant control system by decomposing the matrix into three parts. The first two parts depend on input and time; the third depends solely on the structure of the system model. The theory and algorithm encompassed in the program are developed and explained in Chapters II and III.

Basic assumptions of the algorithm include:

1. The model is a linear, time-invariant, single-input single-output system.
2. The matrix A and the vectors $\underline{B}$, $\underline{C}$, $\underline{x}(t$[illegible]$)$, and $\underline{\text{[illegible]}}$ are real (complex for Program LKPC).
3. Nominal values of A, $\underline{B}$, $\underline{C}$, $\underline{x}$, and [illegible] are good approximations of their true values.
4. All λ_i are distinct and nonzero.
5. Each [illegible]$_i$ appears linearly in A, B, C, and/or $\underline{x}_o$.
6. [illegible]$_i$ may appear more than once and in any location of A, B, C, and/or $\underline{x}_o$.
7. Sample spacing [illegible] is constant.
8. The input U(t) is piecewise constant.

AD-A100 816   AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 12/1
AN INTERACTIVE PROGRAM FOR THE CALCULATION AND ANALYSIS OF THE --ETC(U)
MAR 81 L K PALMER
UNCLASSIFIED AFIT/GA/EE/81M-1 NL

2 OF 2
AD
A100816

END
DATE
FILMED
7 81
DTIC

9. If U(t) is a continuous function, the program user has discretized U(t) over each inverval $[t_{k-1}, t_k]$ for k=1,2,. . ., K.

10. The discretized input $U(t_k)$, k=1, 2, . . ., K, is known exactly.

11. The number of sample times K is at least equal to ($\geq$) the total number of parameters NP.

12. Maximum values which may be used in Program LKP are:

    a. NA (dimension of A) = 10

    b. K (number of samples) = 35

    c. NP (total number of parameters) = 10

    d. NPA, NPB, NPC, NPX (number of parameters in each of A, $\underline{B}$, $\underline{C}$, and $\underline{x}_o$) = 10

    These values, however, may be increased simply by changing the dimensions of the appropriate matrices and vectors in the program.

In explaining the use of the program, each prompt and then the possible user responses are given in the order in which they occur in a computer run. The computer output will be typed in capital letters in this user's guide; the user responses will be typed in lower case letters. If at any time the user wishes to terminate a run, he may enter %A in place of any input. (Due to quirks of the computer, he may have to do this twice.)

After calling Program LKP, the following message will appear:

```
TITLE: NEW ALGORITHM FOR CALCULATING THE
       SENSITIVITY MATRIX OF A LINEAR,
       SINGLE-INPUT SINGLE-OUTPUT (SISO)
       TIME-INVARIANT SYSTEM

       D(X(TIME))/DT=A(P)*X(TIME.P) + B(P)*U(TIME)
       Y(DISCRETE TIME)=C(P)* XO(DISCRETE TIME)

       WHERE XO= X(P,T=0)
```

```
        AND P=THETA, VECTOR OF PARAMETERS OF INTEREST.
        A, XO, B, AND C ARE REAL.

    IF A, XO, B, AND C ARE COMPLEX, TYPE
    %A (RETURN) AND CALL PROGRAM LKPC.

    DO YOU WANT TO KNOW
    HOW TO USE THIS PROGRAM?
    TYPE 1 FOR YES, 2 FOR NO.
```

If the user types 1, definitions of all the vectors, matrices, and other quantities used in the program and an explanation on how to use them will be given.

After the example, or if the user types 2, the prompt

```
ENTER NA, NPA, NPX, NPB, NPC, NP
```

will appear. These values must be input as integers separated by commas since the READ statements are unformatted. The next prompt will be

```
IS THE NOMINAL STATE VECTOR
XO EQUAL TO 0?
   1=YES, 2=NO.
```

If the user responds with 1, the program will set all values of the elements of $\underline{x}_o$ to zero and skip to the next prompt. if the response is 2, the user will be asked to input $\underline{x}_o$:

```
ENTER XO
```

In Program LKP, $\underline{x}_o$ is real; in Program LKPC, $\underline{x}_o$ must be entered in complex notation.

After $\underline{x}_o$ has been entered, the matrices and vectors A, IA, B, IB, IX, C, and IC will be requested:

```
ENTER A, IA, IX, B, IB, C, IC
```

A, $\underline{B}$, and $\underline{C}$ are real in Program LKP (complex in Program LKPC); and A must be entered by columns. $\underline{B}$ and $\underline{C}$ are one-dimensional arrays. IA has dimension 3 x NPA, where the first row contains the numbers of the parameters of A <u>in increasing order</u>; the second row contains the row address of each

parameter, and the column address of each parameter is stored in the third row. IX, IB, and IC have dimension 2 x NPX (or NPB or NPC), where the parameter number is stored in the first row <u>in increasing order</u> and the parameter location in the second row. The matrices IA, IX, IB, and IC are integer arrays and must be input by columns.

Once the system matrices and vectors have been entered, the computer will echo-print them along with the eigenvalues EIG, eigenvectors EIGV, and reciprocal eigenvectors REIGV. The matrices EIG and EIGV are calculated using IMSL subroutine EIGRF for real A or EIGCC for complex A (Ref 6: EIGRF, EIGCC). REIGV is obtained by inverting the transpose of EIGV via IMSL subroutine LEQT1C (Ref 6: LEQT1C). EIG has dimension NA. Both EIGV and REIGV have dimension NA x NA; and each column of EIGV (REIGV) is an eigenvector (reciprocal eigenvector). The eigenvalues are then checked. If any are repeated, the message

```
A MATRIX HAS REPEATED EIGENVALUES.
PLEASE AMEND STATE EQ'NS SO THAT
 A HAS DISTINCT EIGENVALUES.
```

and the program will terminate with the statement

```
STOP      REPEATED E'VAL
```

If any of the eigenvalues equal zero, the message

```
E'VAL CAN'T BE ZERO.
```

will appear, and the program will end.

When all the eigenvalues of A are distinct and nonzero, the prompt

```
WOULD YOU LIKE TO SEE THE CONSTANT
VECTORS?  1=YES, 2=NO.
```

will be given. If 1 is entered, $\underline{v}_j'\underline{x}_o$, $\underline{Cu}_j$, and $\underline{v}_j'\underline{B}$ will be printed for j=1, 2, . . ., NA. By typing a 1 in response to

```
WOULD YOU LIKE TO SEE THE
PARAMETER-SENSITIVE MATRICES?
   1=YES, 2=NO.
```

the user may see $\lambda_{j(i)}$, $\underline{Cu}_{j(i)}$, $\underline{v}'_{j(i)}\underline{x}_o$, $\underline{v}'_{j(i)}\underline{B}$, $\underline{v}'_j\underline{x}_{o(i)}$, $\underline{v}'_j\underline{B}_{(i)}$, and $\underline{C}_{(i)}\underline{u}_j$ for j=1, 2, . . ., NA and i=1, 2, . . ., NP. These values are placed in matrices of dimension NA x NP, in which the ith column contains the values associated with the ith parameter.

If $\underline{x}_o$ is not the zero vector, the structure-dependent part of the sensitivity matrix

$$G = \begin{bmatrix} G_{zi} \\ G_{zs} \end{bmatrix} \tag{19}$$

is calculated and printed. If $\underline{x}_o$ is the zero vector, however, G_{zi} is identically zero; and only G_{zs} is calculated and printed. By responding to the prompt

```
DO YOU WANT A SINGULAR VALUE
DECOMPOSITION OF G=GZI,GZS?
   1=YES, 2=NO.
```

the user may choose to perform a singular value decomposition of G in Eq (19) or of G_{zs} in the case where $\underline{x}_o$ equals the zero vector. The inverse condition number $\frac{1}{\kappa}$ of G (or G_{zs}) is also calculated. The user may then opt to see the right and left singular vectors or only the singular values and the inverse condition number by answering the prompt

```
AFTER TAKING SING. VAL., GTOT=
   UT * S * V
TYPE 1 IF YOU WANT TO SEE
JUST THE SINGULAR VALUES (S)
AND 1/CONDITION NUMBER OF G (CNG)
TYPE 2 IF YOU ALSO WANT TO SEE THE
SINGULAR VECTORS.  ENTER 1 OR 2.
```

After the structure-dependent part of the sensitivity matrix has been calculated, the user receives a message to enter the sample spacing, the number of discrete times, and the discretized input:

```
D=SAMPLE SPACING.  ENTER D.

KIN=NUMBER OF DISCRETE INPUT/OUTPUT TIMES.
ENTER KIN, UP TO 25

U(K),K=1,KIN, ARE DISCRETE INPUTS.
ENTER INPUTS.
```

If the user wishes to change the system matrices and vectors to yield a better matrix G, he may enter "%A" in response to the preceding prompt and begin the program again. If the user enters the values requested, the matrices E, F, and S are calculated. The E and F matrices may be seen by entering 1 in response to

```
DO YOU WANT TO SEE E AND F?
   1=YES, 2=NO.
```

The sensitivity matrix S is always printed, and by typing 1 after the question

```
DO YOU WANT A SINGULAR VALUE
DECOMPOSITION OF SMAT?
   1=YES, 2=NO.
```

the singular value decomposition and inverse condition number of S will be formed. As before, the user may choose to see the left and right singular vectors along with the singular values and inverse condition number of S.

At this point, if the user feels that the sensitivity matrix is too ill-conditioned or otherwise not good enough for his purposes, but the G matrix is satisfactory, he may conveniently change the input, sample spacing, and/or the number of samples by entering 1 in response to

```
DO YOU WANT TO CHANGE JUST THE INPUT?
   1=YES, 2=NO.
```

This feature of the program is particularly helpful when the user has taken too large a sample spacing or not enough samples to adequately discretize an input such as a sinusoid. Then Δ and K may be changed without the necessity of re-entering the system matrices and vectors. The message "D = SAMPLE SPACING...." reappears.

Of course, if the user answers the last question with a 2, the program will end with the message

STOP   NO MORE INPUT, YOU SAID.

An example follows:

```
TITLE:  NEW ALGORITHM FOR CALCULATING THE
        SENSITIVITY MATRIX OF A LINEAR,
        SINGLE-INPUT SINGLE-OUTPUT (SISO)
        TIME-INVARIANT SYSTEM

        D(X(TIME))/DT=A(P)*X(TIME,P) + B(P)*U(TIME)
        Y(DISCRETE TIME)= C(P)* X0(DISCRETE TIME)

        WHERE X0= X(P,T=0)
        AND P=THETA,VECTOR OF PARAMETERS OF INTEREST
        A,X0,B, AND C ARE REAL.
IF A,X0,B, AND C ARE COMPLEX, TYPE
CR (RETURN) AND CALL PROGRAM LXP.

DO YOU WANT TO KNOW
HOW TO USE THIS PROGRAM?
TYPE 1 FOR YES,2 FOR NO.

   47000B    CM STORAGE USED
      3.703 CP SECONDS COMPILATION TIME

                                          2

ENTER NA,NPA,NPX,NPB,NPC,NP
3,2,0,0,2,4
IS THE NOMINAL STATE VECTOR
X0 EQUAL TO 0?
   1=YES,2=NO.
1
ENTER A,IA,IX,B,IB,C,IC
0,-.01,1,-3,1,3,1,2,2,3,0,1,.01,0,3,1,4,2
```

```
A=
0.  1.
-.01  -2.

X0=
0.  0.

B=
0.  1.

C=
.01  0.

IA=
1  2
2  2
1  2

IX=
0
0

IB=
0
0

IC=
3  4
1  2

NP=  4

EIG=
(-.005012562893384,0.)
(-1.994987437107,0.)

EIGV=
(15.94878930485,0.)  (-.5041324665147,0.)
(-.07994430946381,0.)  (1.005737941324,0.)

REIGV=
(.06285862133278,0.)  (.004996519341488,0.)
(.03150827928217,0.)  (.9967993315529,0.)
WOULD YOU LIKE TO SEE THE CONSTANT
VECTORS?   1=YES,  2=NO.
1
REIGVX=  (0.,0.)  (0.,0.)
IEIGV=  (.1594878930485,0.)  (-.005041324665147,0.)
REIGVB=  (.03150827928217,0.)
(.9967993315529,0.)
WOULD YOU LIKE TO SEE THE
PARAMETER-SENSITIVE MATRICES?
   1=YES,2=NO.
1
```

```
EIGI=
(.5025189076296,0.) (-.00251890762906,0.)
(0.,0.) (0.,0.)
(-.5025189076296,0.) (1.00251890763,0.)
(0.,0.) (0.,0.)

CEIGVI=
(-.04027472046678,0.) (.0002018795693531,0.)
(0.,0.) (0.,0.)
(.001273061789178,0.) (-.002539742276072,0.)
(0.,0.) (0.,0.)

REVIX0=
(0.,0.) (0.,0.) (0.,0.) (0.,0.)
(0.,0.) (0.,0.) (0.,0.) (0.,0.)

REVIB=
(-.007956636182366,0.) (.01587338922545,0.)
(0.,0.) (0.,0.)
(-.2517170029174,0.) (.001261747308457,0.)
(0.,0.) (0.,0.)

REVX0I=
(0.,0.) (0.,0.) (0.,0.) (0.,0.)
(0.,0.) (0.,0.) (0.,0.) (0.,0.)

REVBI=
(0.,0.) (0.,0.) (0.,0.) (0.,0.)
(0.,0.) (0.,0.) (0.,0.) (0.,0.)

DIEIGV=
(0.,0.) (0.,0.) (15.94879930485,0.)
(-.07994430946391,0.)
(0.,0.) (0.,0.) (-.5041324685147,0.)
(1.005737941324,0.)

GZI=

(-.002537974280957,0.) (.002537974280957,0.)
(.5025189076296,0.) (-.002518907629606,0.)

(.002537974280957,0.) (-.002537974280958,0.)
(-.5025189076296,0.) (1.00251890763,0.)

(.0025252525252,0.) (-.00001265798710449,0.)
(0.,0.) (0.,0.)

(.0025252525252,0.) (-.0050378470634,0.)
(0.,0.) (0.,0.)
SINCE X0=0,GZI=0
DO YOU WANT A SINGULAR VALUE
DECOMPOSITION OF GZI?
   1=YES,2=NO.
1
```

```
AFTER TAKING SING.VAL.,GZS=
   UTS * S * VS
TYPE 1 IF YOU WANT TO SEE
JUST THE SINGULAR VALUES (S)
AND 1/CONDITION NUMBER OF GZS
(CNGZS),TYPE 2 IF YOU ALSO
WANT TO SEE THE SING. VECTORS
ENTER 1 OR 2
1

S=
1.1483066004 .4376285279265 .005770277272927 .002199122691648
CNGZS= .001915100628074
D=SAMPLE SPACING.  ENTER D.
.1
KIN=NUMBER OF DISCRETE INPUT/OUTPUT TIMES.
ENTER KIN, UP TO 30
10
U(K),K=1,KIN ARE DISCRETE INPUTS.
ENTER INPUTS
1,1,1,1,1,1,1,1,1,1
 DO YOU WANT TO SEE E AND F?
   1=YES,2=NO.
1
E=

(.9994998693186,0.) (.8191412498907,0.)
(.09994998693186,0.) (.08191412498907,0.)

(.9989979897692,0.) (.6709923872725,0.)
(.1997995979539,0.) (.1341984774545,0.)

(.9984973612258,0.) (.5496375427775,0.)
(.2995492093677,0.) (.1648913628333,0.)

 .9979969835628,0.) (.4502307837776,0.)
(.3991987934251,0.) (.1800923113511,0.)

(.9974968566544,0.) (.3688026069629,0.)
(.4987484283272,0.) (.1844013034814,0.)

(.9969969803749,0.) (.3021014284305,0.)
(.5981981882225,0.) (.1812608570583,0.)

(.9964973545989,0.) (.2474637416783,0.)
 .6975481482192,0.) (.1732246191748,0.)

 .9959979792005,0.) (.2027077758661,0.)
(.7967983833604,0.) (.1621662069288,0.)

 .9954988540545,0.) (.1660462867921,0.)
(.8959489868649,0.) (.1494416581129,0.)

 .9949999790354,0.) (.1360153629025,0.)
 .9949999790354,0.) (.1360153629025,0.)
```

```
F=

(.09997494137406,0.)  (.0906565859741,0.)
(.004998329561204,0.)  (.004382213553039,0.)

(.1999997822376,0.)  (.1649171351198,0.)
(.01998663839246,0.)  (.01539792035473,0.)

(.2997745476976,0.)  (.2257470141645,0.)
(.04495491266831,0.)  (.03050432809717,0.)

(.3995992628483,0.)  (.2755752773159,0.)
(.07989314609091,0.)  (.04786143613181,0.)

(.4993739527715,0.)  (.3163916630736,0.)
(.1247913398851,0.)  (.06616099787756,0.)

(.5990986425363,0.)  (.3498260483192,0.)
(.1796395027934,0.)  (.08449436228299,0.)

(.6987733571994,0.)  (.3772135324396,0.)
(.2444276510707,0.)  (.1022507257287,0.)

(.7983981218048,0.)  (.3996477504116,0.)
(.3191458084796,0.)  (.1190391172724,0.)

(.8979729613841,0.)  (.4180345437622,0.)
(.4037840062853,0.)  (.1346238606403,0.)

(.9974978009561,0.)  (.4330777338265,0.)
(.4983722932505,0.)  (.1489043914555,0.)

IMATI=

(3.8493841572138E-8,0.)  (.00001509555675404,0.)
(.004662644977645,0.)  (.07046311138759,0.)

(5.69837666326E-7,0.)  (.00001095970021239,0.)
(.01757944161566,0.)  (.164229017063,0.)

(.000002673662106997,0.)  (.00003363479775958,0.)
(.03720023528557,0.)  (.2255605456455,0.)

(.000007942395867407,0.)  (.00007863950362337,0.)
(.06238439772961,0.)  (.2752628721525,0.)

(.00001779352801419,0.)  (.0001225157490837,0.)
(.09195206033453,0.)  (.315930747533,0.)

(.00007435727671822,0.)  (.0002647033365594,0.)
(.125264191748,0.)  (.3491931516799,0.)

(.0000921298143093,0.)  (.000287391746339,0.)
(.161534415763,0.)  (.3744155529487,0.)
```

```
(.00009450940316449,0.) (.0004082775762095,0.)
(.2003796010494,0.) (.3986433350587,0.)

(.0001415317117218,0.) (.0005347460371404,0.)
(.2411831545419,0.) (.4168155980313,0.)

(.0002019521526587,0.) (.0006760184848679,0.)
(.2836318061266,0.) (.4316560109697,0.)
DO YOU WANT A SINGULAR VALUE
DECOMPOSITION OF SMATS?
   1=YES,2=NO.
1
DO YOU WANT TO CHANGE JUST THE INPUT?
   1=YES,2=NO.
1
AFTER TAKING SING. VAL.,
   SMATSR=UTRANS * S * V

TYPE 1 IF YOU WANT TO SEE
JUST THE SINGULAR VALUES (S)
AND 1/CONDITION NUMBER OF SMATS (CNS)
TYPE 2 IF YOU ALSO WANT TO SEE THE
SINGULAR VECTORS.  ENTER 1 OR 2.
1


S=
1.1108753223438 .1506809533672 .000071097099820737 .0000068877438914
CNS= .000006135199315
D=SAMPLE SPACING.  ENTER D.
.1
KIN=NUMBER OF DISCRETE INPUT/OUTPUT TIMES.
ENTER KIN, UP TO 30
5
U(K),K=1,KIN ARE DISCRETE INPUTS.
ENTER INPUTS
1,1,1,1,1
 DO YOU WANT TO SEE E AND F?
   1=YES,2=NO.
2

SMATS=

 (3.3499941572156E-8,0.)  (.0000015199555375404,0.)
 (.00422224977643,0.)  (.0163311399755,0.)

 (5.6993766938E-7,0.)  (.000010959700812339,0.)
 (.01797444161566,0.)  (.164339017163,0.)

 (.000002697368810997,0.)  (.00011363477975556,0.)
 (.02720023525557,0.)  (.3356054736455,0.)

 (.0000073433956974007,0.)  (.00017343363623317,0.)
 (.04623349777391,0.)  (.3793497319735,0.)

 [illegible]
 [illegible]
```

```
DO YOU WANT A SINGULAR VALUE
DECOMPOSITION OF SMATS?
   1=YES,2=NO.
1
DO YOU WANT TO CHANGE JUST THE INPUT?
   1=YES,2=NO.
2
AFTER TAKING SING. VAL.,
   SMATSR=UTRANS * S * V

TYPE 1 IF YOU WANT TO SEE
JUST THE SINGULAR VALUES (S)
AND 1/CONDITION NUMBER OF SMATS (CNS)
TYPE 2 IF YOU ALSO WANT TO SEE THE
SINGULAR VECTORS.  ENTER 1 OR 2.1


S=
.524129955744 .03441741786263 .000008131953288623 3.615396783183E-7
CNS= 6.897301452798E-7

STOP  NO MORE INPUT (0 STATE)
```

Vita

Lieutenant Linda Kneen Palmer was born in Burlington, Vermont, in 1956. She received a B. A. in Mathematics from St. Michael's College in Winooski, Vermont. Lt Palmer is married to Leslie Allan Palmer. They have one daughter, Heather Marie.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GA/EE/81M-1	2. GOVT ACCESSION NO. AD-A100 826	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) An Interactive Program for the Calculation and Analysis of the Parameter Sensitivities in a Linear, Time-Invariant System		5. TYPE OF REPORT & PERIOD COVERED MS Thesis
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Linda K. Palmer, 1st Lt, USAF		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT) Wright-Patterson AFB, OH 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE March 1981
		13. NUMBER OF PAGES 97
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release; IAW AFR 190-17

Frederic C. Lynch

21 MAY 1981

FREDERIC C. LYNCH, Major, USAF
Director of Public Affairs

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Parameter Sensitivities
Output Sensitivity Matrix
Linear, Time-Invariant Control System
Parameter Identifiability
Parameter Estimation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

In this paper, a new algorithm is developed to calculate the output "sensitivity matrix" of a linear, time-invariant, single-input single-output control system with piecewise constant input and output measurements taken at constant time intervals. The algorithm incorporates the singular value decomposition to investigate parameter identifiability and estimation accuracy in relation to the system as a whole and in relation to the model of the system. As a result, a structural condition on iden-

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. tifiability is imposed; and the system designer now has a tool to evaluate how well the model describes the system.

The algorithm is verified by checking its results with those using a standard software package for numerical integration. It is then used to investigate input and structural design issues.

END

DATE
FILMED
7-81

DTIC